Lossless Compression of Structured and Unstructured Multi-View Image Data

von Buelow, Max

(2020)

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Master's Thesis

Lossless Compression of Structured and Unstructured Multi-View Image Data

Maximilian Alexander von Bülow

May 2019

Technische Universität Darmstadt
Department of Computer Science
Graphisch-Interaktive Systeme

Supervisor: Dr. rer. nat. Stefan Guthe

Declaration of Authorship

I certify that the work presented here is, to the best of my knowledge and belief, original and the result of my own investigations, except as acknowledged, and has not been submitted, either in part or whole, for a degree at this or any other university.

Darmstadt, May 27, 2019

Maximilian Alexander von Bülow

Abstract

Photometric multi-view 3D geometry reconstruction and material captures are important techniques for cultural heritage digitalization. Capturing images of these datasets with high resolutions and high dynamic range and store them using the proprietary raw image format of the camera enables future proof application of this data. As these images tend to consume immense amounts of storage, compression is essential for long time archiving. In this thesis, I present multiple approaches for compressing multi-view and material reconstruction datasets with a strong focus on data created from cultural heritage digitalization. These approaches address different types of redundancies occuring in these datasets and are able to compress datasets with arbitrary resolutions, bit depths and color encodings. The individual approaches are further evaluated against each other and state-of-the-art image and file compression algorithms. The approach with highest compression efficiency archieves rates from 1.77:1 to 2.09:1 compared to an uncompressed representation for multi-view datasets and 2.75:1 for a material capture dataset. Compared to the PNG algorithm, it archieves compression rates of 1.33:1 in average on both dataset types.

Keywords (according to ACM CCS): [Computing methodologies]: Computer graphics—Image compression, [Hardware]: Communication hardware, interfaces and storage Signal processing systems—Digital signal processing, [Computing methodologies]: Computer vision—Computer vision problems—Reconstruction

Zusammenfassung

Photometrische 3D Geometrierekonstruktionen und Materialaufnahmen sind wichtige Techniken für die Digitalisierung kulturellen Erbes. Die Einzelbilder dieser Datensätze mit einer hohen Auflösung und einem hohen Dynamikumfang aufzunehmen und sie in einem von der Kamera produzierten rohen Dateiformat zu speichern erlaubt eine zukunftsfähige Weiterverarbeitung dieser Daten. Da diese Bilder dazu neigen immensen Speicherplatz zu verbrauchen, ist die die Kompression dieser Bilder ist für die Langzeitarchivierung essentiell. In dieser Arbeit präsentiere ich mehrere Ansätze zur Kompression von Datensätzen für die 3D-Rekonstruktion und Materialaufnahme in der Digitalisierung kulturellen Erbes als Anwendungsgebiet. Diese Ansätze addressieren verschiedene Arten von Redundanzen, die in diesen Datensätzen auftreten und sind fähig diese Datensätze mit beliebigen Bildauflösungen, Bit-Tiefen und Farbkodierungen zu kodieren. Des Weiteren werden diese Ansätze gegenseitig und gegen andere dem Stand der Technik entsprechenden Bild- und Dateikompressionsverfahren verglichen. Der die höchste Kompressionseffizienz erreichende Ansatz komprimiert 3D-Rekonstruktion mit Raten von 1.77:1 bis 2.09:1 und Materialaufnahmen mit 2.75:1. Verglichen mit dem PNG-Algorithmus werden Kompressionsraten von durchschnittlich 1.33:1 erreicht.

Titel: Verlustfreie Kompression Strukturierter und Unstrukturierter Multi-View-Bilddaten

Contents

1 Introduction

The compression approaches presented in this thesis aim to reduce the storage size of multi-view and material capture datasets by compressing them losslessly. These approaches vary in terms of computational effort, algorithmic complexity, targeted type of redundancy and type of capturing environment (structured or unstructured). They do not make restrictions on the resolution, nor the bit depth of the individual images as long as they are consistent within the dataset.

Recent work mainly focused on video compression that is similar to multi-view image compression as multi-view images can be arranged in a seqential order. Unfortunately, most of these approaches perform lossy compression, i.e. removing redundancies that are not visible to the human perceptional system. Algorithms that are specialized for multi-view image compression often depend on additional disparity data that is usually computed by geometry reconstruction algorithms, but as disparities also need to be encoded, additional storage is required. Thus, I use basic techniques for video and still image compression that are further extended towards the different properties of multi-view datasets. Compressing material capture datasets was not part of active research during the past years.

In this thesis, I first illustrate basic techniques of redundancy reduction and continue with compositions of these, adapting optimally to the properties of the data. In chapter 4, results from these approaches are evaluated and discussed not only with main focus on the resulting file size but also on computation time and simplicity of the algorithms used. Sub-optional performing approaches are further evaluated under theoretical aspects to show why they are failing.

Best performing approaches can be used to reduce image files sizes of digitalized cultural heritage datasets and enable efficient archiving and transfer of these.

1.1 Motivation

Digitalization of cultural heritage is important to enable easy access to cultural artifacts. Additionally, these artifacts get occasionally destroyed due to political conflicts, accidents

or natural disasters. However, having a digital copy of its geometrical structure and its material properties enables present and future generations archeological analyses of these artifacts, even though its original version is not present anymore [San+14].

These digital copies, called datasets, consist of a set of images captured from different camera positions for multi-view geometry reconstructions and with different lighting conditions for material captures along with the extrinsic and intrinsic parameters for each camera and the light source positions. To archieve the goal of transfering artifacts accross generations, it is important to store these datasets at multiple secure locations. This is currently limited by the enormous image file sizes that are caused by capturing the images with high resolutions and high dynamic range making storage and transfer inefficient and expensive. In order to reduce storage requirements and transfer times, compression of multi-view and material capture datasets is mandatory. To keep the original quality, the datasets need to be compressed losslessly. In the past, research on cultural heritage digitalization focused on creating reconstructions instead of on efficient storage of the datasets.

Currently, the individual images of these datasets are stored independently in their camera's raw file format, that already contains inter-pixel redundancy compression. Due to hardware limitations on the cameras, these compression algorithms yield non-optimal compression rates. Industry cameras often transfer pixel data using a bus system to a standard computer system in a uncompressed form. These images are often directly written out to uncompressed file formats due to interfaces that do not offer advanced features like compression. Sometimes, Gzip or similar dictionary compression algorithms are then used to compress these uncompressed files, unfortunately also yielding non-optimal compression rates. Dictionary coders theoretically are able to reduce redundancies between views, but they are limited by their dictionary size.

1.2 Related Work

In this section, the related work on compression of multi-view datasets is dividied into lossy, near-lossless and lossless techniques. Further, this techniques are categorized into the ones that do not require additional disparity information (depth or motion vectors) or the ones that do. Techniques that require additional disparities usually have the disadvantage that depth values need to be encoded in order to decompress the multi-view image data, yielding sub-optimal compression rates.

Lossy Compression Starting with approaches that require additional previously estimated disparity information, Gelman et al. [GDV10] developed a prediction scheme that

divide each view into a layer based representation whose layers have approximately constant depth values followed by a 3D wavelet transformation across the viewpoint and spatial dimensions of corresponding layers. The wavelet coefficients are encoded using an arithmetic coder as the backed. Velisavljević et al. [Vel+11] used in their lossy "texture+depth" approach the depth values to perform a depth-image-based rendering for obtaining a prediction. Furthermore, a dynamic bit-rate allocation is used as the compression backend. Perra and Assuncao [PA16] created a pseudo video sequence to encode the light field data, which usually has only a small camera baseline, with the h.265/HEVC video encoder, effectively creating a motion field using the Block Motion Compensation algorithm to obtain temporal relations. This archieves better compression rates than compressing the views using the JPEG image format. The work of Gehrig and Dragotti [GD07] does not make use of disparity information by using a quadtree with the assumption that views are modeled using a piece-wise 2D polynomial function. Their compression backend is also dynamic bit-rate allocation. Siegel et al. [Sie+97] described background principles of compressing 3D stereoscopic videos.

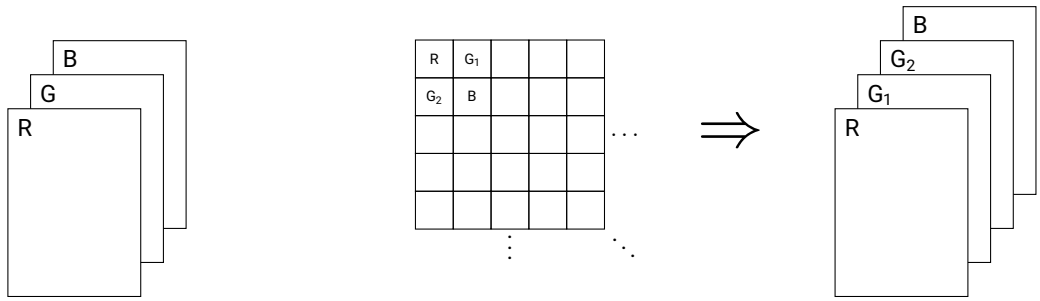
Near-lossless compression Aydinoglu et al. [AKH95] presented a near-lossless approach that also makes use of previously estimated disparity values that are encoded lossy in the bitstream and interpolate the view using a bi-directional disparity estimator and a modified version of the *Subspace Projection Technique*. In this work, an arithmetic coder was used as the compression backend. To avoid error propagation, some frames are encoded independently, similar to the work of Battin et al. [BVL10].

Lossless compression The lossless approach by Martins and Forchhammer [MF98], sometimes denoted by "LOCO-3D" as it is a 3D extension to the LOCO-I algorithm by Weinberger et al. [WSS96], compresses video frames using a set of different predictors that are sequentially evaluated to optimally predict the neighboring and motion compensated pixels from reference frames. As this algorithm was designed for video compression, a pseudo video sequence must be created prior to the actual encoding. Brunello et al. [Bru+02] extended this to a mathematically well-posed algorithm, called "LOPT-3D", that uses a weighed sum of previously coded pixels by solving a linear equation system resulting in better compression rates than LOCO-3D. LOPT-3D uses an Golomb-Rice coder as its coding backend. Carotti and De Martin [CD05] used multi-frame Block Motion Compensation in the CALIC framework to further enhance the results of the LOPT-3D approach. Kamisetty and Jawahar [KJ03] estimated three view relationships using a trilinear tensor to create a prediction followed by a residual coding step. Perra [Per15] encoded

multi-view images by minimizing the entropy of the 1D Differential Pulse-Code Modulation (DPCM) and encoding the block-DPCM data using the LZMA dictionary coder backend. A comparison between DPCM, 3D Discrete Cosinus Transformations (DCT) and Priniciple Component Analysis (PCA) predictions based on estimated disparity values was done by Shah and Dodgson [SD01].

2 Background

Multi-view 3D geometry reconstruction and material capture datasets usually consist of a set of images and their camera and light parameters. These images are two-dimensional arrays of any color representation depending on the camera sensor. Colors are usually represented in the standard RGB (sRGB) color space using three color values for preprocessed images or using the Bayer Pattern with vendor specific bit lengths for unpreprocessed (raw) images. The Bayer Pattern can be seen as a single pixel with four color channels: red, two green channels and a blue one. The color representations are visualized in fig. 2.1.



(a) *sRGB layers. The red, green and blue channel are separated.*

(b) *The bayer pattern. Groups of the four different types of pixels are transformed into a four-layer representation similar to the sRGB encoding.*

Figure 2.1: *Color channel representations.*

These sets of images contain different statistical redundancies that can potentially be compressed losslessly:

- Temporal/inter-view redundancies: Pixels from correlating regions (i.e. pixel refering to the same point in an object) in two images usually have similar values.
- Spatial/inter-pixel redundancies: Adjacent pixels usually have similar values.
- Spectral redundancies: Adjacent frequency coefficients usually have similar values.
- Coding redundancies: Redundancies caused by inefficient coding of symbol representations.

The spatial and spectral redundancies are caused by the heavy tailed image statistics of natural images [WS00]. Temporal redundancies are caused by multiple images capturing the same object but with different capturing conditions. Coding redundancies occur by allowing trivial random access of the pixels, because files and memory can usually only be addressed in multiples of 8 bit.

This chapter describes some basics about geometry reconstructions, material captures and techniques that are later used to reduce redundancies occurring in the datasets of these techniques.

2.1 Geometry Reconstruction

In geometry reconstruction, a set of input images is used to reconstruct a three-dimensional scene. The input images must be taken under fixed lighting conditions from different camera positions and scene objects should remain static for a sequential capture. Geometry reconstruction is archieved by dividing the whole process into the following steps.

1. Bundling: Joint determination of all intrinsic and extrinsic camera parameters and sparse reconstruction of keypoints (keypoints are described in section 2.9) exploiting the epipolar constraints. If the camera parameters are known, this step is not required.

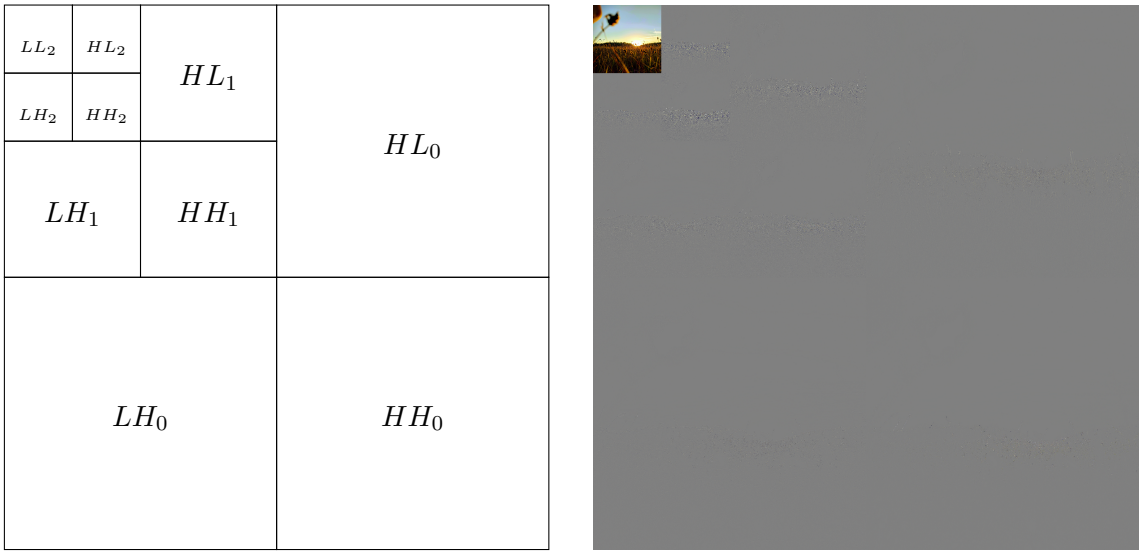
2. Dense reconstruction (*Multi-View Stereo*, MVS): Depth values are reconstructed for each pixel, again exploiting the epipolar constraints. This can, e.g., be done by the plane-sweeping algorithm introduced by Collins [Col96]. Afterwards, the pixels can be transformed to the 3D scene using their reconstructed depth values.

3. The resulting dense point cloud can be used to perform a surface reconstruction.

Each captured image of a geometry reconstruction together with the intrinsic and extrinsic camera parameters forms a *view*. Extrinsics of the camera can either be structured, meaning they are fixed for each reconstruction and arranged in a defined way, or unstructured.

2.2 Material Capture

Input images for material capture have in addition to a set of fixed camera parameters, a variable light position for each view. The positions can also be structured or unstructured. A position on the surface that has different radiance samples along different light source



(a) *The subbands of a wavelet transformation.* **(b)** *An example wavelet transformed image.*

Figure 2.2: *Structure of a wavelet transformation.*

directions is called a *Lumitexel*. The parameters of a *bidirectional reflectance distribution function* (BRDF) model of the Lumitexels can be estimated using a non-linear least squares solver and compared against databases of known material parameters. A common reflectance model is the Lafortune model [Laf+97]. Redundancies occuring in material capture datasets are, due to the fixed camera position, that each input image shows the same object but with different radiance values.

2.3 Discrete Wavelet Transformation

The discrete wavelet transformation (DWT) of a one-dimensional signal x is computed by passing it through a series of low and high pass filters and downsampling steps resulting in two subbands. The low pass filtered values are the significant coefficients l and the high pass filtered values are the detail coefficients h. An advantage over the Fourier transformation is that wavelet transformations captures frequency and spatial information at the same time.

Instead of a combination of filters and downsampling, the wavelet transformation can be expressed using a polyphase matrix and a signal split into its odd and even indexed parts x_o, x_e, the polyphase components of the signal. This reduces the computational effort, as the downsampling takes place before the filtering and avoids discarding filtered coefficients.

The polyphase matrix

$$P(z) = \begin{bmatrix} H_e(z) & G_e(z) \\ H_o(z) & G_o(z) \end{bmatrix} \tag{2.1}$$

is constructed using the odd and even polyphase components of the low and high pass filters H_e, H_o, G_e, G_o. The wavelet transformation can now be expressed by:

$$\begin{bmatrix} s \\ d \end{bmatrix} = P(z) \begin{bmatrix} x_o \\ x_e \end{bmatrix} \tag{2.2}$$

The polyphase matrix is now factorized into

$$P(z) = \begin{bmatrix} s & 0 \\ 0 & \frac{1}{s} \end{bmatrix} \prod_{i=1}^{M} \begin{bmatrix} 1 & U_i(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -P_i(z) & 1 \end{bmatrix} \tag{2.3}$$

to obtain the lifting scheme, where $P_i(z)$ are the so called prediction steps and $U_i(z)$ the update steps. The scaling factor s is used to preserve the following energy term:

$$\sum_i |x_i|^2 = \sum_i |h_i|^2 + \sum_i |l_i|^2 \tag{2.4}$$

A variant of the discrete wavelet transformation is the formulation of Cohen et al. [CDF92]. As an example, the filters of the CDF 5/3 (also called LeGall 5/3) are

$$H = \{-\frac{1}{8}, \frac{2}{8}, \frac{6}{8}, \frac{2}{8}, -\frac{1}{8}\}$$
$$G = \{-\frac{1}{2}, 1, \frac{1}{2}\}$$

and the factorization of the polyphase matrix leads to the following prediction and update steps:

$$h_i = x_{2i+1} - \frac{1}{2}(x_{2i} + x_{2i+2})$$
$$l_i = x_{2i} + \frac{1}{4}(h_{i-1} + h_i)$$

The discrete wavelet transformation is usually executed successively multiple times on the low pass subbands, such that multiple resolutions, called subbands, are created.

The extension of the wavelet transformation to 2D signals (e.g. images) transforms

rows and columns of the signal separately. First, all rows are transformed independently using a standard 1D wavelet transformation, which results in rows of significant and detail coefficients w_l, w_h. Subsequently, the coefficients from the prior step are transformed column-wise using an 1D wavelet resulting in the significant and detail coefficients of w_l, namely w_{ll} and w_{lh} and the significant and detail coefficients of w_h, namely w_{ll} and w_{lh}. Figure 2.2 shows the wavelet subbands and a wavelet transformed example image.

2.4 Arithmetic Coding

Arithmetic coding is a near-optimal way of lossless entropy coding. The basic concept of arithmetic coding is to successively reduce an interval of cumulated frequencies. Using this intermediate state it is effectively possible to encode symbols with fractional amount of bits. The coded size yields approximately the entropy of the input. In this section, I describe the b-bit integer based algorithm of Moffat et al. [MNW98], which makes use of the prefix property of Rissanen and Langdon [RL79] stating that no code of a symbol should be a prefix of another.

The internal state of the arithmetic coder consists of L, R and o, where $[L, L+R)$ is the current interval of the coder and o the number of outstanding bits, which is described later. Each symbol $s_i, i = 0, ..., N$ has a range $[l_i, l_i + f_i)$ of cumulated frequencies computed by all previous frequencies $l_i = \sum_{j=0}^{i-1} f_j$. The sum of frequencies is given by $t = l_N + f_N$.

The interval of the coder is initialized with $L = 0$ and $R = 2^{b-1}$. Initially, to encode a symbol x_i, the scaling factor $r = \frac{R}{t}$ between the coder range and the total frequency must be computed. Then, the left boundary of the coder range is re-computed by adding the scaled left boundary of the symbol $L \Leftarrow L + r \cdot l_i$. The right boundary of the coder is re-computed by $R \Leftarrow r \cdot f_j$.

Under the aspect that R decreases after each coding step of a symbol, it cannot be represented anymore after several steps. Additionally, L will continuously grow and reach the maximal representable integer value. Thus, a re-normalization step is introduced as long as R is smaller than 2^{b-2}. First, a check is performed whether 2^{b-1} is smaller, greater or inside the coder interval. If it is smaller, the arithmetic coder writes a 0 to the bitstream in order to singalize that case to the decoder. If it is greater, the arithmetic coder writes a 1 and reduces L by 2^{b-1}, which shifts the Interval to the left-hand side and prevents an eventual integer overflow. If it is inside the interval, it cannot be signaled to the decoder on how to handle this case unambiguously and thus this bit will be handled as *outstanding* by incrementing o. This decision can only be taken for the next unambiguous case and then coded as the inverse bit of that future step. In principle, this procedure writes a value

slightly over or slightly under 0.5 to the bit stream. Additionally, L can safely be reduced by 2^{b-2} as $L + R > 2^{b-1}$ and $R \leq 2^{b-2}$ hold by construction. Finally, L and R are scaled by a factor of two.

According to Shannon [Sha01] the total number of bits h required for encoding a symbol s_i is defined in eq. (2.5). The total amount of bits, the entropy H, is then defined as the expected value of h with respect to the frequencies f_i (eq. (2.6)). The symbol frequencies f_i are proportional to their probabilities $p(s_i) = \frac{f_i}{t}$.

$$h(s_i) = \log_2 \frac{1}{f_i} \tag{2.5}$$

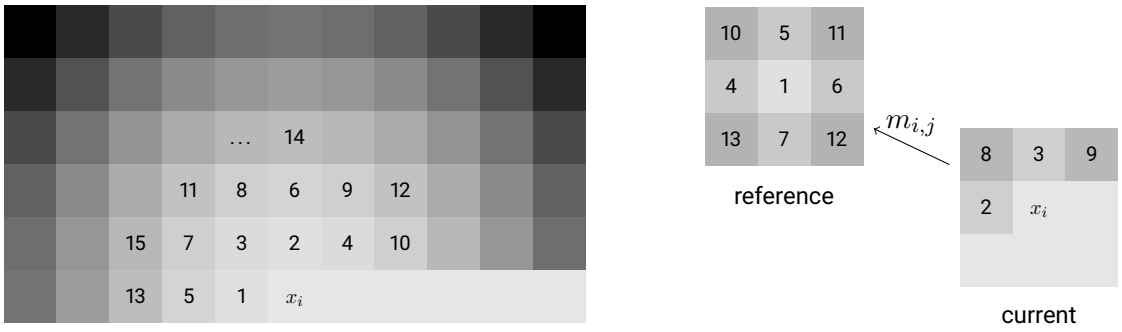
$$H(s) = \sum_i f_i h(s_i) = -\sum_i f_i \log_2 f_i \tag{2.6}$$

The unit managing the frequencies is called the *Model*. In principle, the frequencies can be modeled without restrictions and can be static or dynamic for each symbol. An example of dynamic models are adaptive models. Adaptive models count the number of symbols while coding, whereby the coder will use different frequency ranges for each coding step. The entropy of a dynamic model converges to the entropy of a static model with the final frequencies assumed to be known [MNW98]. Because the coder is based on cumulated frequencies, it makes sense to store them directly instead of re-compute them for each coding step. Fenwick [Fen93; Fen95] introduced an efficient frequency table for dynamic models, that has logarithmic access and manipulation times. A model is *Context-Adaptive* if it consists of multiple frequency tables that are chosen on basis of previously coded values. The whole process is then usually called *Context-Adaptive Arithmetic Coding* (CAAC).

2.5 Optimal 3D Prediction

LOPT-3D is a lossless video compression scheme by Brunello et al. [Bru+02] with reasonable computational effort. In the first step of the algorithm a motion search is performed. Afterwards, a linear optimization on basis of the previously coded pixels and motion compensated pixels of the reference image is performed to find the optimal weighting for pixels adjacent.

First of all, they define an indexing order for the adjacent pixels in the current frame. It corresponds to a circle, defined by eq. (2.7), around the current pixel that only includes pixels that are already encoded. Further, a second indexing is defined that corresponds to a sphere, defined by eq. (2.8), around the current pixel in the already coded part of the current frame, i.e. the first layer, and the motion compensated frame as the second layer.



(a) *2D indexing.* **(b)** *3D indexing.*

Figure 2.3: *Indexing order of the adjacent pixels [Bru+02].*

The pixels relative to the corresponding motion vector $m_{i,j} = m_{\lfloor \frac{x}{16} \rfloor, \lfloor \frac{y}{16} \rfloor}$ are used for the second layer. The distance between both layers used for the indexing is chosen as $\frac{1}{4}$. Both indexing orders are stated in fig. 2.3.

$$i_{2D} = \sqrt{(x_0 - x)^2 + (y_0 - y)^2} \tag{2.7}$$

$$i_{3D} = \begin{cases} \sqrt{(x_0 - x)^2 + (y_0 - y)^2}, & \text{current image} \\ \sqrt{(x_0 - (x - m_{i,j}))^2 + (y_0 - (y - n_{i,j}))^2 + \frac{1}{4}}, & \text{reference image} \end{cases} \tag{2.8}$$

The notation $x_{-i,-j}$ stands for the i^{th} pixel with respect to the two dimensional indexing (eq. (2.7)) and from there the j^{th} pixel with respect to three dimensional indexing (eq. (2.8)). Given N denoting the order of the predictor, the prediction is calculated as follows.

$$\hat{x} = \sum_{j=1}^{N} a_j x_{0,j} \tag{2.9}$$

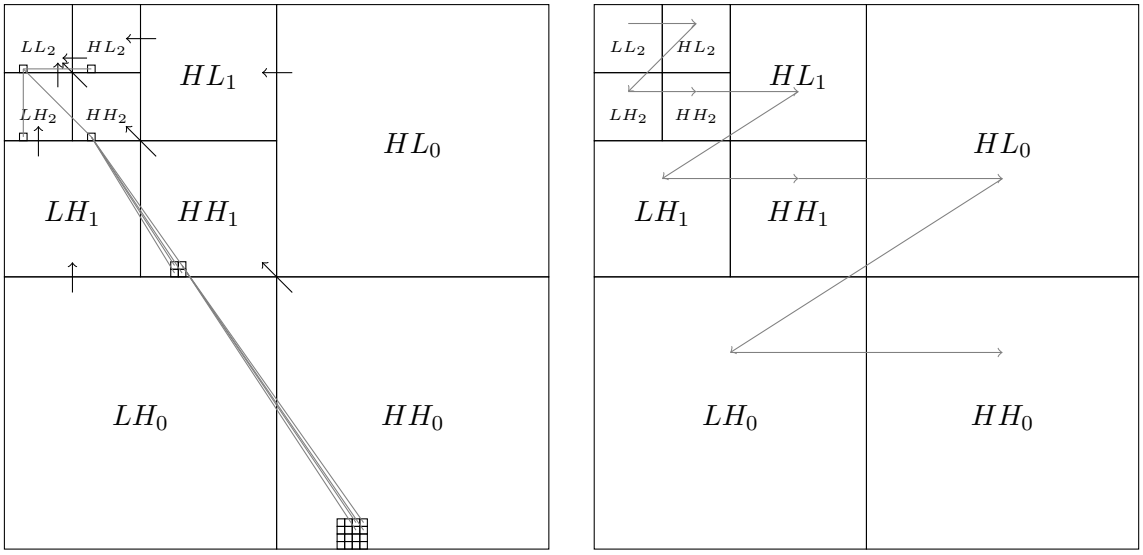
To minimize the squared error of the prediction, they define $C = (x_{-i,-j})_{i=1,...,M;j=1,...,N}$ and $X = (x_{-1,0}, ..., x_{-M,0})^T$, where M is the size of the circle of pixels to be taken into account and solve the linear equation system $Ca = X$. The solution of this equation system is according to Brunello et al. [Bru+02] given by $a_0 = \left(C^T C\right)^{-1} C^T X$. However this solution tends to create artifacts in very homogeneous or extremely heterogeneous regions. During the implementation of the algorithm, it pointed out that a direct approach using a complete orthogonal decomposition (COD) tends to be much more numerically stable.

The residual of the prediction is coded using a Golomb-Rice coder. For the motion

search they use a block size of 16 pixels and a search radius of 8 pixels. Furthermore they set $N = 7$ and $M = 70$, which means that they chose two pixels (the upper and the left one) from the current layer and five pixels (the center one and its four neighbors) from the reference layer for the weighting (Figure 2.3(b)). The original implementation also uses caching techniques for the weighting vector to further reduce the computational effort. However, this is not further discussed here since computers tend to have enough computational power nowadays and archiving is not a time critical application.

It is also important to mention, that this compression scheme does not have a lossy analogon, because the coding residual would continuously propagate during compression. Thus, it is mandatory to transmit the actual residual image.

2.6 Embedded Zerotree Wavelet Coding



(a) *Tree dependencies [Sha93]* **(b)** *Traversal order [Sha93]*

Figure 2.4: *The topological relations between the different subbands. Each coarse subband has four children in the finer subband, except of the coarsest subband. The traversal order is defined to traverse coarse subbands first and then successively each finer subband. This ensures that parents are coded prior to children.*

Embedded Zerotree Wavelet Coding (EZW) is a lossy compression technique introduced by Shapiro [Sha93] that introduces a topological tree relationship between wavelet subband resolutions and encodes this tree using an arithmetic coder. The principle behind

zerotrees is, that this topological relationship exploits the self similarity of the signal. More specific, they exploit the fact that insignificant coefficients, i.e. coefficients that are close to zero, remain insignificant in finer subbands at the corresponding position. Thus, these coefficients do not need to be encoded.

Shapiro [Sha93] defines a tree structure and the order of traversal for the compression, as visualized in fig. 2.4, as follows. Each parent coefficient of a coarse subband has, due to the successive two-dimensional subsampling of the signal, four children in the next finer subband. The coefficients in the coarsest subband, however, only have three children, one in each subband. For coding, a breath-frist traversal is chosen, such that the significant subband is coded first, followed by coding all three corresonding detail subbands descending into each level. This ensures that the parents are coded prior to the children.

Although lossless compression cannot make the assumption that coefficients close to zero do not have to be encoded, the compression can still take advantage of the fact that the absolute value of child-coefficients rarely exceed the absolute value of their parent coefficient.

2.7 The YCoCg-R Color Space

The YCoCg-R color space [MS03] consists of a luma channel and a green and orange chroma channel. As the human has greater sensitivity to light differences than to color differences, some image formats internally convert to a luma-chroma color space and do higher lossy compression on the chroma channels. However, these formats are always converting back to sRGB resulting in a high correlation between the color channels. This section introduces the reversible and lossless YCoCg-R color scheme in order to be able to decorrelate these images again. This would be useful for intermediate formats originating from lossy formats such as undistorted versions that are usually encoded losslessly. The lossless conversion between sRGB and YCoCg-R is stated in the following lifting scheme.

$$
\begin{aligned}
C_o &= R - B \\
t &= B + \frac{C_o}{2} \\
C_g &= G - t \\
Y &= t - \frac{C_g}{2}
\end{aligned}
$$

2.7.1 Bayer Pattern Extension

The YCoCg-R color space can be extended to work on images that are captured with the bayer pattern and thus having four color channels R, G_1, G_2 and B. The basic idea behind that is to compute the average A_g between both green channels G_1 and G_2, apply the standard YCoCg-R transformation and encode the difference ΔG between the average green channel and G_2 anlong with Y, C_o and C_g. This leads to the following lifting scheme.

$$\begin{aligned}
\Delta G &= G_2 - G_1 \\
A_g &= G_1 + \frac{\Delta G}{2} \\
C_o &= R - B \\
t &= B + \frac{C_o}{2} \\
C_g &= A_g - t \\
Y &= t - \frac{C_g}{2}
\end{aligned}$$

However, experiments on real data in the raw camera format pointed out, that this transformation does not improve rates of lossless compression.

2.8 Block Motion Compensation

The term *motion compensation* describes algorithms that create a prediction $\hat{x}^{(k)}$ over the video frame $x^{(k)}$ using its previous frame $x^{(k-1)}$ and the temporal relations between both. The reconstruction of these relations is called *motion search*.

Moreover, in *Block Motion Compensation* (BMC) the frames are subdivided into blocks $B_{i,j}$ of fixed size (usually 16×16 pixels). For each block of frame $x^{(k)}$ a motion vector $(m_{i,j}, n_{i,j})$ is searched with in the search space s in frame $x^{(k-1)}$ that minimizes (eq. (2.11)) a distance metric (usually the sum of squared distances, SSD: eq. (2.10)).

$$d_{i,j}(m,n) = \sum_{(x,y)\in B_{i,j}} \left(x^{(k)}_{x,y} - x^{(k-1)}_{x-m,y-n} \right)^2 \tag{2.10}$$

$$(m_{i,j}, n_{i,j}) = \underset{m,n\in\left[-\frac{s}{2},\frac{s}{2}\right)}{\operatorname{argmin}} d_{i,j}(m,n) \tag{2.11}$$

2.9 Image Stitching

Image stitching is, e.g., used to create panorama pictures from multiple overlapping images. A base assumption, described by Adelson and Bergen [AB91], is that the camera position must be fixed and only the rotation of the camera is allowed to change. If the camera position changes, as it is the case for multi-view datasets (Section 2.1), the stitching still returns an affine transformation that tries to superimpose the corresponding image features.

The image stitching algorithm searches in its first step all keypoints and their descriptors in a pair of images. This is for example done by the *Scale-Invariant Feature Transform* (SIFT) algorithm. In the SIFT algorithm, the keypoints correspond to the extremal values of the Difference-of-Gaussians (DoG) that occur on multiple resolution levels. Afterwards, keypoints with low contrast surroundings are discarded and the ratio $R = \frac{tr(H)^2}{\det(H)}$ is calculated to estimate the curvature at the remaining keypoints. Keypoints with low curvature are discarded to increase the stability of the descriptors. The descriptors of SIFT keypoints are orientation histograms with bin size 8 of a 16×16 pixel region, that is subdivided into four 4×4 pixel regions, resulting in a descriptor length of $4 \cdot 4 \cdot 8 = 128$ numbers.

Now, after determining keypoints and their descriptors, the algorithm searches in both images for each descriptor its pairwise nearest neighbor based on the descriptor value. The nearest neighbor is called a match. Afterwards, a homography H is calculated from the matches by stacking up the rows

$$\begin{bmatrix} 0 & 0 & 0 & x & y & 1 & -xy' & -yy' & -y' \\ -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \end{bmatrix} \tag{2.12}$$

for each match $H \cdot (x, y)^T = (x', y')^T$ and decompose it using a *Singular Value Decomposition* (SVD) into USV^T. The homography then corresponds to the right singular vector $(V_{3i+j,9})_{i,j=1,2,3} = H$. In practice, the *Random Sample Consensus* (RANSAC) algorithm is used to repeatedly pick four random samples and find the homography with the maximum number of inliers. An inlier is defined as a transformed keypoint that has a difference $|H \cdot (x, y)^T - (x', y')^T|$ to its corresponding match smaller than a certain threshold.

2.10 Predictive Coding

Most lossless compression schemes depend on predictive coding introduced by Elias [Eli55]. Predictive coding creates a prediction p_i over a value x_i using prior information. This makes decoding the values possible, as the decoder is able to compute exactly the same prediction

as the encoder. The encoder, then only encodes the prediction error (or residual)

$$e_i = m_i - p_i \tag{2.13}$$

that should by design only contain small values with less entropy. The decoder can then reconstruct by transforming eq. (2.13) to

$$m_i = p_i + e_i. \tag{2.14}$$

Given a set of values $x = \{x_0, ..., x_N\}$, a prediction for value x_i can be computed using all prior coded values $\hat{x}^{(i)} = \{x_j; j < i\}$. Thus, a predictor is an arbitrary function $p_i = P(\hat{x}^{(i)})$, that depends on all previously coded values. A simple linear predictor, for example, uses a weighted sum

$$p_i = P(\hat{x}^{(i)}) = \sum_{j<i} a_j \cdot x_j \tag{2.15}$$

as a prediction function.

The prediction error is usually encoded using an arithmetic coder to reduce the coding redundancies of the error values that have less entropy.

2.11 GNU Zip

GNU Zip (Gzip) is a commonly used dictionary compression algorithm. It is based on the DEFLATE algorithm that is a combination of LZ77 [ZL77] and Huffman Coding [Huf52]. LZ77 compresses the input by finding the longest prefix of the current input in a fixed amount of recent data, the *dictionary*. If a prefix is found, the offset to it and its length is encoded. If not, the plain character must be encoded, the input is advanced by that character and the prefix search is repeated.

The Huffman Code is similar to arithmetic coding as both are entropy coders. A fixed binary sequence is assigned to each symbol where its lenght is inversely proportional to the probability of the symbol.

2.12 Portable Network Graphics

The Portable Network Graphics image compression format [Bou97] compresses images using the DEFLATE algorithm (Section 2.11) with a prior decorrelation step.

The decorrelation is applied by a filter that is chosen from a predefined set of filters to minimize the resulting file size. The filters consist of the difference of the current pixel p with the left neighbor $p-a$ or the upper neighbor $p-b$, the difference between the current pixel, the average of the upper and left neighbor $p - \lfloor \frac{a+b}{2} \rfloor$ and the Paeth filter [Pae91] that chooses the difference to a, b or c, whichever is closest ot $a+b-c$. The difference is then encoded using predictive coding (Section 2.10). Adjacency relations are visualized in fig. 2.5.

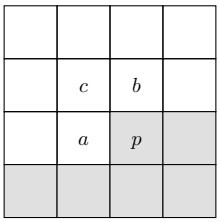


Figure 2.5: *The PNG algorithm's adjacency relationships of p. a is the left neighbor of p, b the upper neighbor and c the diagonal (upper left) neighbor. The gray shaded part is outstanding for enconding. The part without shading are pixels that are encoded already and thus can be used for predictions.*

3 Methodology

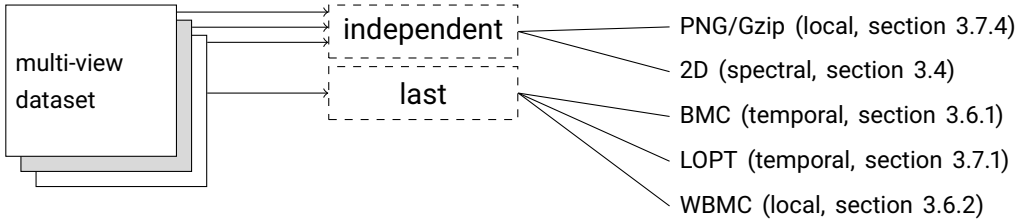
As mentioned in chapter 2 multi-view and material capture datasets consist of different types of redundancies. This chapter describes multiple approaches that address the different redundancies to compress the given datasets. Sections 3.1 to 3.3, define the file header format, the preprocessing of each individual image of the dataset and how they are ordered. Continuing with the main part, section 3.4 describes approaches that reduce spectral redundancies, followed by the residual coding techniques in section 3.5 that partially depend on these. Section 3.6 focuses on temporal redundancy reduction and section 3.7 on a combination of local and temporal redundancies. Chapter 3 shows the data dependencies and an overview of all approaches.

3.1 File Header

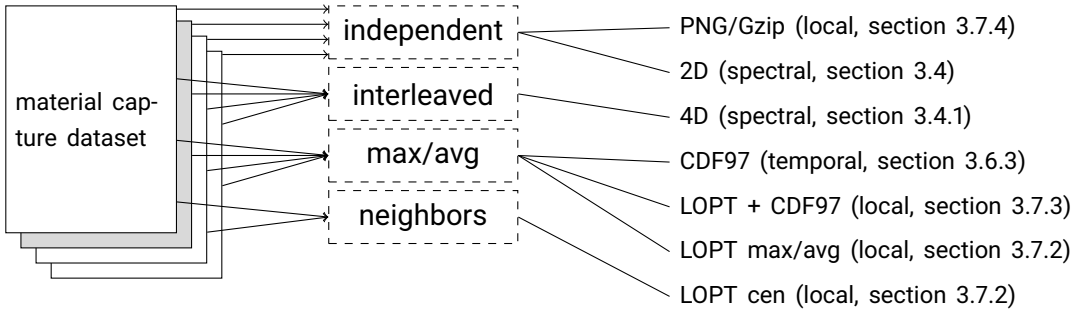
The decompression algorithm requires the same meta information (e.g. the width and the height of the images for performing 2D wavelet transformations) as the encoder. The file format includes a leading header structure encoding meta information about the dataset, required for correct decompression. The first value is a magic number with ASCII interpretation "`MVC\0`", making identification of the file format possible. The following two values are unsigned 32 bit integers representing the width and the height of the individual images, followed by an unsigned 32 bit integer that represents the number of images contained in the dataset. The three remaining unsigned 8 bit values represent the number of color channels, the bitdepth and, if applicable, the number of wavelet transformation levels that is used for spectral approaches and defaults to four. As this structure is only 19 B large, it is encoded in an uncompressed form.

3.2 Image Preprocessing

The individual color channels of the sRGB color space correlate to each other [GP02]. Thus, if the number of color channels of the images in the dataset is exactly three, a lossless



(a) *Overview and data dependencies of multi-view compression approaches.*



(b) *Overview and data dependencies of material capture compression approaches.*

Figure 3.1: *Overview and data dependencies of the compression approaches presented in this chapter. The gray shaded box marks the current picture to be compressed. The type of data dependency or operation is stated in the dashed box.*

conversion into the YCoCg-R color space (Section 2.7) is performed. This decorrelates the individual color channels into a luma and two chroma channels. Each chroma channel tends to have less entropy than the luma channel, making further compression more efficient.

3.3 Sequences

The compression rate depends on the order of images, since closer images tend to have higher correlations. For structured material capture datasets, the algorithm uses a path that goes back and forth along the successively moving arc of the capturing device (Figure 3.2). Unstructured geometry reconstructions use Jarnik's algorithm [Jar30] (also called Prim's algorithm) to create a topological tree structure from the set of nearest neighbors. This can be explained by closer views tending to have smaller motion vectors. The tree is stored for the decompression phase by writing the list of parent node indices to the output stream.

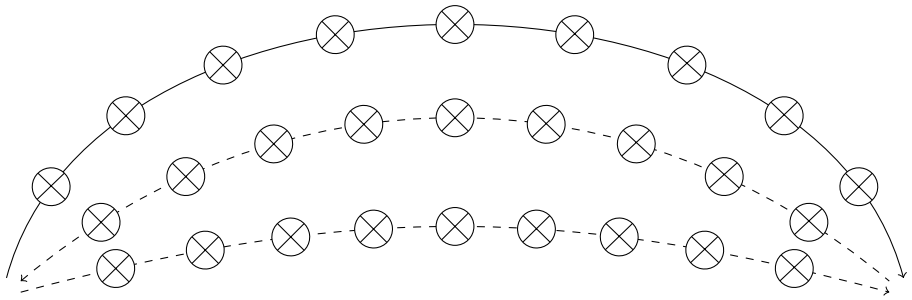


Figure 3.2: *Sequencing of structured material capture datasets. The circles with crosses represent for an image that was captured with the light source turned on at the respective position. The arrows represent the moving arc (only three arc positions were drawn for a less cluttered visualization).*

3.4 Spectral Redundancies

In order to remove spectral redundancies, a lossless CDF 5/3 wavelet transformation (Section 2.3) with the number levels from the header structure (by default four) is executed first. Using the tree structure defined in section 2.6, a context selection is performed based on the parent coefficient of the wavelet transformed image. Therefore, the parent coefficient is divided bitwise into two parts at the floored halve of the bitdepth b limited to eight $n_b = \min\{\lfloor \frac{b}{2} \rfloor, 8\}$ and the most significant bits (MSB) are used as the bin index for the context selection switching between $m = 2^{n_b}$ models that capture statistics of the actual coefficient's n_b MSB. All these models are based on dynamic frequency tables. The actual coefficient's n_b MSB are then used to perform a second context selection of further m models resulting in a total number of $m^2 + m$ models. Afterwards, the selected model is used to encode the remaining least significant bits (LSB) of the actual coefficient. Assuming a bitdepth of 8 bit, the number of models to be allocated is $16^2 + 16 = 256 + 16$. For a bitdepth of 16 bit $256^2 + 256 = 65536 + 256$ models need to be allocated. The whole process is visualized in fig. 3.3. A detailed analysis how often each model was actually used during compression can be found in section 4.3.3.

The second model selection ensures that the model adapts faster to the coefficients to be compressed. As the root of the wavelet tree, i.e. the coarsest subband, has no parent coefficient, it is encoded using one further independent model. The limit of eight bits for the bin index is used to prevent excessive memory requirements that are caused by the exponential hierarchy of the models.

The encoding is done in the traversal order described in section 2.6 to ensure that the

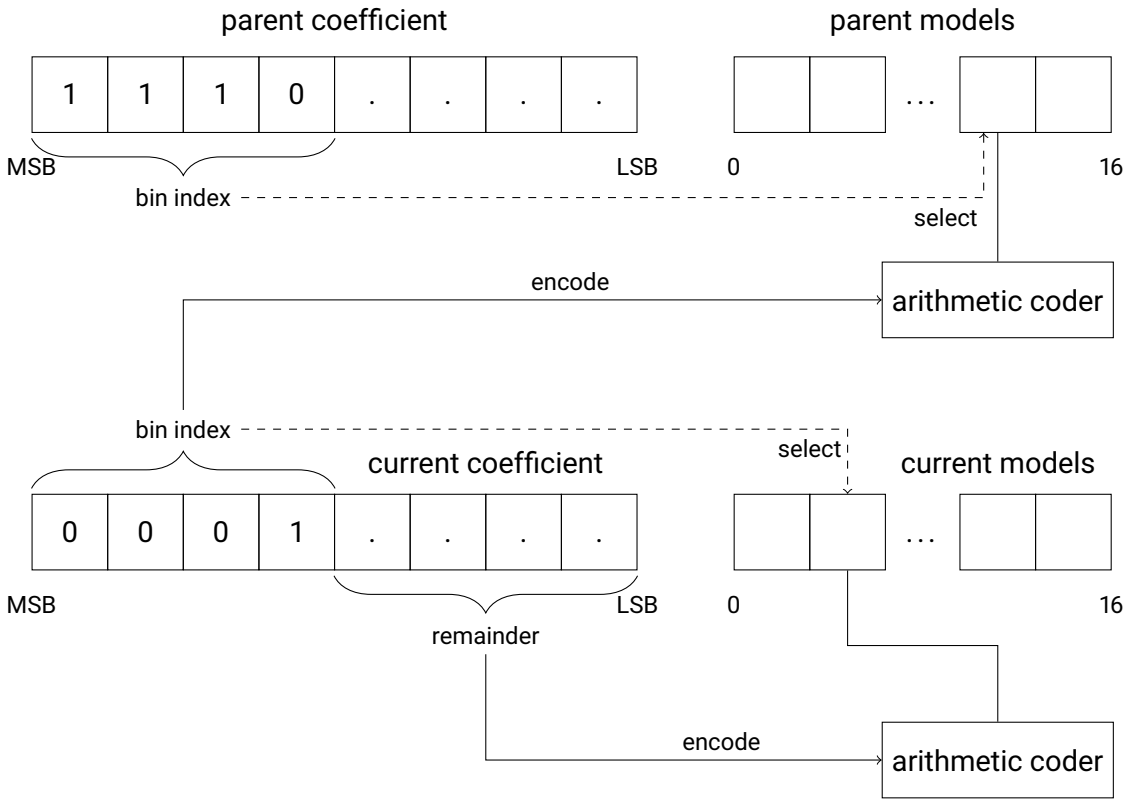


Figure 3.3: *Flow chart of the context-adaptive coding of the current coding exploiting the correlations to the topological parent coefficient. Bits used for model selection are marked using a dashed arrow and then connected to the arithmetic coder using a solid line. Bits that are encoded are marked using a solid arrow.*

parent coefficient is encoded prior to the children coefficients. The bin index used for the second model selection is available to the decompression algorithm by directly decoding it from the stream.

This two-dimensional approach can be used for compression of material capture datasets as well as multi-view datasets since it makes no assumptions about correlations between views. This approach will be extended to multiple dimensions in section 3.4.1 to also capture inter-view redundancies.

3.4.1 Multidimensional

Because eliminating spectral redundancies in a two-dimensional manner was very successful during the work of this thesis, I decided to check whether they can also be exploited between views. These higher dimensional wavelet transformations can only be used for material

capture datasets as multi-view capture datasets have no chance of inter-view redundancies without any preprocessing to find correlating pixels. Pixels on the same positions from different views are generally independent of each other and a preprocessing (e.g. using block motion compensation) is not worth to implement, since the wavelet transformation is done in one single step while BMC is based on a step-by-step compression of all frames resulting in an extremely complex implementation. Furthermore, the compression rate benefit is expected to be low. Executing a high dimensional wavelet transformation on residual images of a simple BMC prediction leads to the same problem that views do not correlate, because residual images violate the natural image assumtion of the wavelet tree.

The first two dimensions are, as before, the spatial dimensions of the image. The following dimensions represent the different views and its structure. More specifically, the hemisphere structure of the material capture device allows a two dimensional representation of the light source position. The wavelet transformation is then applied simultaneously in all dimensions. The wavelet tree structure remains similar, except that now each parent has 16 instead of 4 children. The traversal order remains analogous: finer subbands are coded after coarser ones.

3.5 Residual Coding

The previously presented spectral approaches do not create residual images as they do not rely on predictions and are thus working directly on the input images. In contrast to that, approaches that eliminate local redundancies or approaches that create lossy predictions always depend on residual images to encode the image losslessly. These residual images also need to be encoded efficiently to the bitstream. In this section different techniques for residual image compression are presented. In chapter 4, the different residual coding techniques on each approach are evaluated.

In some cases a prediction is not possible and the actual image must be encoded directly. I assume that this actual image is an residual image from a zero-prediction (meaning an image containing only zeros) and apply the same algorithms.

3.5.1 Arithmetic Coding

A simple technique to encode the residual image is to use an arithmetic coder with an adaptive frequency table as its model. The frequency table has a size of 2^b elements, where b is the bitdepth of the residual image. As the bitdepth is usually not greater than $16\,\text{bit}$, the size of the frequency table would not exceed $65\,536$ elements, which can be easily

handled. The frequency table stores the frequencies of color values used for the arithmetic coder. This approach removes the coding redundancies from the residual images.

3.5.2 Spectral Coding

Despite the fact that residual images violate the natural image property, an spectral coding would be an interesting comparision, especially for cases where no prediction can be computed. To evaluate the spectral redundancies and remove them, the approach from section 3.4 is used.

3.5.3 Gzip

As a comparison, Gzip is used to compress the residual images to remove eventual redundant sequences in the image data. Gzip assumes the image to be a byte stream where each pixel is represented by a combination of multiple bytes. Each color channel is encoded sequentially and independent from each other. As described in section 2.11, Gzip keeps track of previously encoded bytes and avoids re-encoding prefixes by encoding only its offsets to the prior encoding.

3.6 Temporal Redundancies

As described in chapter 2, the datasets may contain inter-view redundancies. This section concerns on these temporal redundancies and makes foundations for the local approaches that are introduced in section 3.7, which eliminate temporal redundancies in its initialization step.

3.6.1 BMC View Prediction

Originally used for video compression, a well known technique to eliminate temporal redundancies is the block motion compensation as described in section 2.8. Since the motion vectors, which result from the motion search tend to be bigger for multi-view datasets than for video compression, a prior image stitching step is applied as described in section 2.9 to estimate a homography between both views that minimize the overall reprojection error. This homography is used to generate a set of motion vectors that is used as an initialization for the actual motion search with a block size of 16 pixels and a search space of 256×256 pixels around the initially motion compensated position. Compared to that, the video compression standard h.264 computes a camera pan vector that is similar to the computation of a homography but has only two degrees of freedom. After all motion

vectors are estimated, the algorithm generates a motion compensation that is used as the prediction of this approach. The motion vectors are encoded by an arithmetic coder with two separate models for horizontal and vertical components.

The standard implementation of the BMC algorithm produces artifacts caused high frequency variations on block boundaries. To reduce the artifacts, overlapping blocks can be used in the motion compensation algorithm with linear interpolation in overlapping parts. Hoever, it pointed out that interpolation of overlaps do not improve compression rates [Gut04].

A motion search within that search space size is extremely computationally expensive but independent for each block, meaning it is an *embarrassingly parallel* problem. Thus, I parallelized the motion search using OpenMP to reduce computation time.

3.6.2 BMC View Prediction in Wavelet Domain

As initial approaches used spectral coding (Section 3.4) to encode the residual images but further research revealed that the BMC algorithm tends to introduce further spectral variations on block boundaries, this approach applies the BMC algorithm in the wavelet domain. Like the BMC approach described in the previous section, the algorithm performs a prior global cumulated motion estimation using the image stitching algorithm to initialize the motion vectors. To archieve compression in the wavelet domain, the algorithm then applies a CDF 5/3 wavelet transformation on the image pair, consisting of the current image and its topological parent. Now, the algorithm uses the topological relations from section 2.6 to scale the current motion vector according to the wavelet level $l = 0, ..., l_{max}$ by the factor $\frac{1}{l+2}$ such that motions from coarser levels correspond to motions from finer levels (i.e. it uses a scaling of $\frac{1}{2}$ for the finest level and continue with $\frac{1}{4}$, $\frac{1}{8}$, ... for the coarser ones). Using fractional pixel steps is common practice in video compression: the result of the motion search is usually refined by $\frac{1}{4}$ pixel steps, the so called *qpel*. As the motion vectors now contain fractional amount of pixels, a bi-linear interpolation is used to get the values between integral pixel positions. The motion vectors are then chosen to minimize the sum of squared differences of the interpolated pixel values on each level of the wavelet tree. To optimize the running time of this approach, the algorithm divides the motion search into two steps with a different search space size and number of wavelet levels included in the search. In the first step, like for the standard BMC, the full search space of 256 pixels is used, but the wavelet levels is limited to the significant and it's corresponding three detail subbands. The second step refines the previous result with a reduced search space of 64 pixels but incorporating all wavelet levels.

After estimating the motion vector in the wavelet domain, the algorithm applies motion

compensation using the motion vectors on each wavelet subband independently. Again, it scales the motion vectors for each wavelet subband respectively and uses bi-linear interpolation to get the actual pixel values. This motion compensation step theoretically does not solve the problem with artifacts on block boundaries, but since the wavelet transformation is performed first, the block boundaries do not hurt.

In the end, the context-adaptive zerotree wavelet coding algorithm described in section 3.4 is used to compress the difference between the current image and the motion compensated wavelet transformation.

3.6.3 CDF 9/7 Radiance Prediction

As described in section 2.2, material captures consist of images with different luminance values, captured from the same viewing position and angle. Thus, pixels only differ in their radiance values, however objects they are showing are the same. This approach assumes, similar to the local approach presented in section 3.7.2, that fine subbands of the different radiance samples are similar to fine samples of the maximum radiance.

This approach is structed as follows. First, high frequency information is discarded from the maximum image and each individual image. Afterwards, a factor image is created from the maximum image with and without high frequencies, describing in a multiplicative way what information is lost when high frequencies are discarded. This factor image is then used to restore the low frequency versions each individual image. The data flow of this approach is visualized in fig. 3.4.

To compute the factor image, a maximum image m is calculated from all radiance samples. This maximum image is then transformed using a CDF 9/7 wavelet transformation with 4 levels. Now, all coefficients from detail subbands are discarded (i.e. set to zero) and an inverse CDF 9/7 wavelet transformation is applied. The resulting image is an approximation m_w of the maximum image containing only the low frequency proportions. Afterwards, the approximation m_w is divided by the original maximum image m to create the factor image f, which contains factors between 0 to 2. This factor image is used to reduce parts where an approximation over-estimates values and enhances under-estimated parts. The original maximum image must be encoded to the bitstream in order to reconstruct the factor image in the decompression counterpart of this algorithm.

As mentioned before, high frequency information is also discarded for each individual image. This is done the same way as for the maximum image. Then the prediction is computed as the factor image multiplied by the low frequency versions of the images. To be able to decompress the dataset, the significant coefficients are sufficient to encode. Computing and encoding a residual image (Section 3.5) afterwards enables lossless decompression of

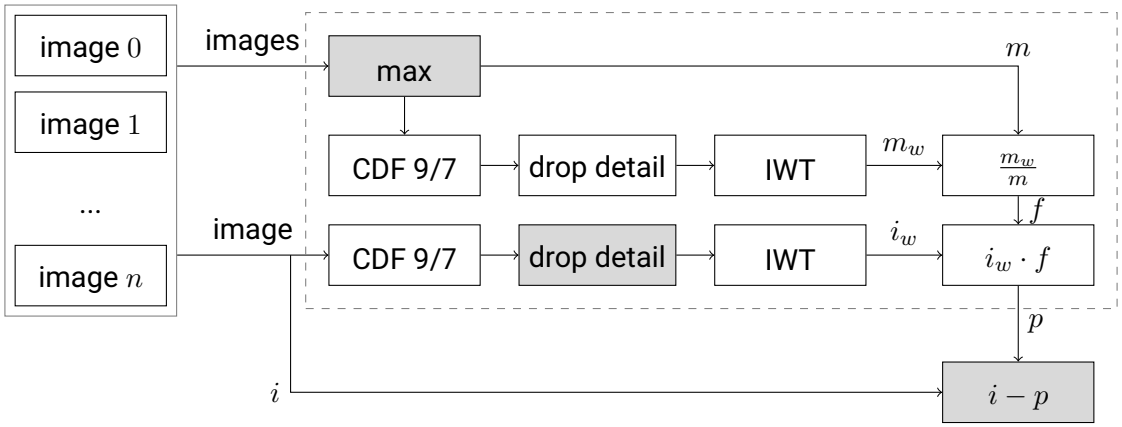


Figure 3.4: *Flow chart of the CDF 9/7 predictor. The dashed box includes the actual predictor. The results of the gray shaded components need to be further processed and encoded.*

the dataset.

Initial experiments used the h.264 video compression standard to encode the maximum image and the approximations of the different radiance samples. During the work on this thesis it turned out, that h.264 is not efficient enough to encode sequences of images losslessly, as it is designed to compress visual redundancies on video sequences with small baselines. Additionally, implementations of h.264 are very inflexible in controlling reference frames as required for the topological tree structure of the data and are not capable of encoding images with a bit depth higher than eight. Thus, I discarded the use of h.264 in the following and implemented the main part of h.264, the Block Motion Compensation algorithm, in a specialized form as previously described in section 3.6.1.

3.7 Local Redundancies

In section 2.5 the LOPT-3D algorithm that reduces spatial and temporal redundancies was presented. This algorithm is used in this section to create predictions for material capture and multi-view datasets with a prior BMC step. I decided against the multi-BMC algorithm of Carotti and De Martin [CD05] as it would add further complexity to the already very costly standard pairwise BMC algorithm. The LOPT-3D algorithm transform local redundancies into coding redundancies by generating a residual image using predictive coding. Adopted from the application in video compression, LOPT-3D depends on a pair of consecutive images: the previous (i.e. the frame that corresponds to the topological parent from section 3.3) and the current one. As mentioned in the previous chapter, the

aproaches presented in this section extend the temporal approaches and thus rely on a prior BMC step. In this section, the term *prediction* is used for the motion compensated image from the prior BMC step.

3.7.1 LOPT View Prediction

As described in section 2.5, a prior motion search (Section 2.8) step is applied to estimate the motion vectors. The motion vectors are then used to learn the weighting of corresponding regions in the image pairs using the LOPT-3D algorithm. The result of it is used as the prediction for this approach. This implementation is similar to the original description of the LOPT-3D algorithm, except that motion vectors are initialized using the image stitching algorithm. A big search space of 256 pixels is chosen, making LOPT-3D capable of compressing datasets that contain higher baselines or rotations than video streams.

3.7.2 LOPT Radiance Prediction

First experiments revealed that the LOPT-3D algorithm estimates a multiplicative weighting of the neighborhood in both layers given the previously coded pixels. Those weights do not have to be necessarily similar but constant along that neighborhood. As luminance changes are also multiplicative factors to the maximum image of all radiance samples, this approach uses the maximum image as a prediction for LOPT-3D. The preceding BMC step is omitted as no motion need to be estimated for material capture datasets. As a comparison to the maximum, I also predict the current image using the average between all radiance samples. In both cases, the maximum or respectively the average, must be encoded beforehand to make the decompression possible. The algorithm uses the spectral compression from section 3.5.2 to encode these predictions in an initial coding step.

Both previous approaches use all images from the dataset to create the maximum or average prediction. These predictions get the more deccorelated to each individual image the bigger the dataset becomes. A more local behaving approach could be to use the mean of the previous and next image as a prediction for the LOPT-3D algorithm. As the decoder needs to be able to replicate this process, that pair of previous and next image must be encoded into the stream before this point. The algorithm uses the spectral compression approach from section 3.5.2 to encode every second image and then predicts every image in-between using the LOPT-3D algorithm on the mean of both adjacent images as its prediction, which is similar to so called *B-frames* in video compression. This process is visualized in fig. 3.5.

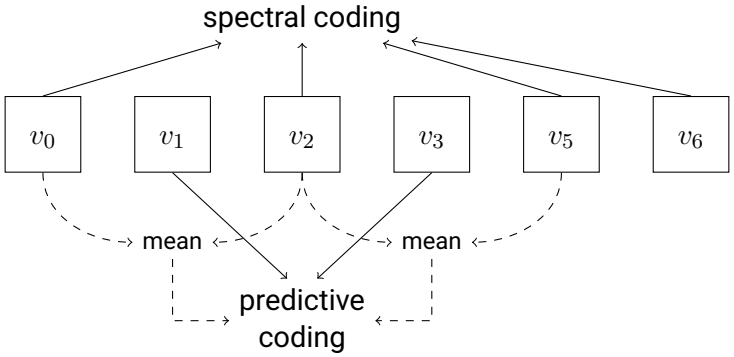


Figure 3.5: *Coding scheme of the mean prediction. In the first step, every second view starting from the first one is encoded using a spectral coder. In the second step, the views in-beween can be predicted using its previous and next nieghbor.*

3.7.3 LOPT-3D with CDF 9/7 Prediction

Results from the CDF 9/7 prediction described in section 3.6.3 still contain local redundancies, since the prediction only targets inter-view redundancies. To enhance the results of the CDF 9/7 prediction by further reducing local redundancies, this approach applies a subsequent LOPT-3D step. Here, the result created from the CDF 9/7 prediction scheme is used as the input for the LOPT algorithm. As before, the preceding BMC step is omitted because material capture datasets do not contain any camera motion.

3.7.4 Gzip and PNG

PNG and Gzip also perform local redundancy reduction. I use Gzip and PNG as state-of-the-art comparisons to our presented approaches. For Gzip, the image is encoded in a row-major format for each color channel independently. Samples are encoded using either 8 bit integers for bit depths smaller or equal than 8 and 16 bit for bit depths greater than 8 (also called *uncompressed encoding* in the following).

PNG is not capable of inter-view redundancy reduction. Gzip is limited to its dictionary size that is too small compared to the image dimensions to capture inter-view redundancies.

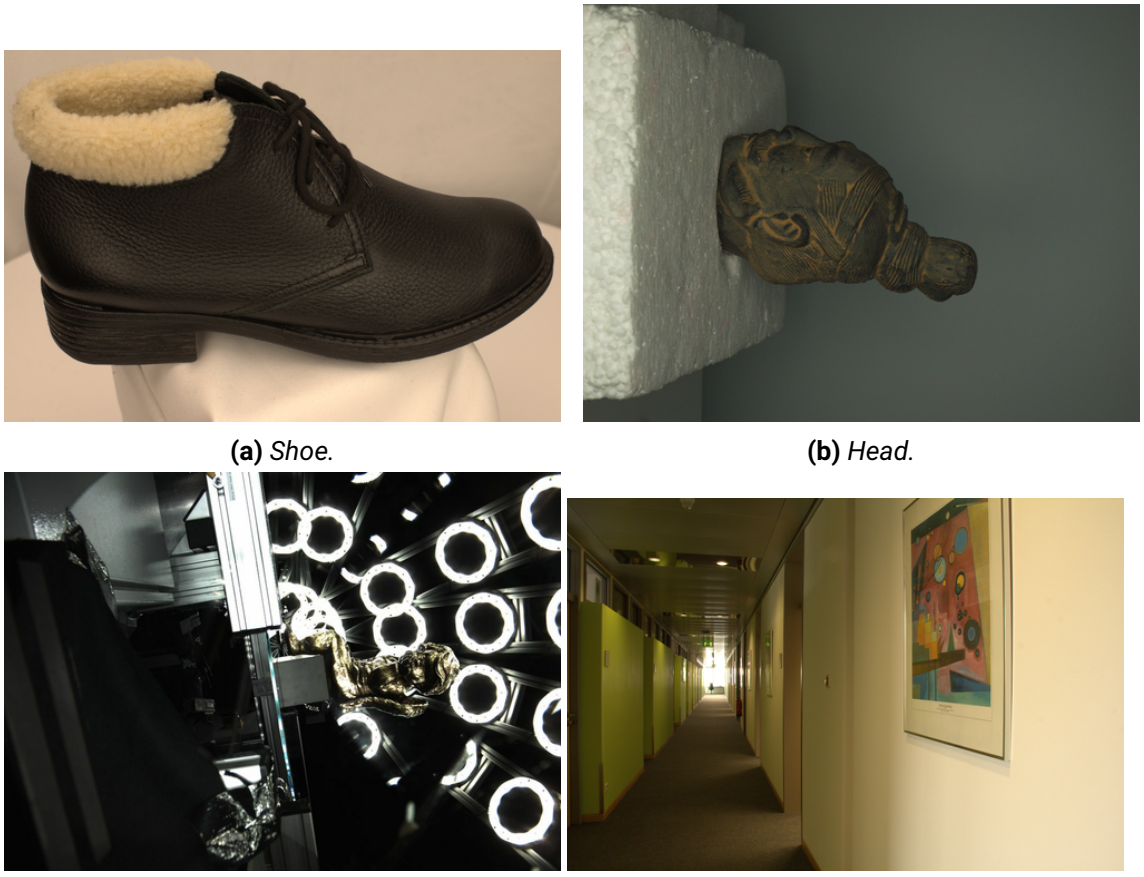
3.8 Selective Predictions

The compression rates of the individual approaches may vary across image regions. Switching between these algorithms for each region individually can make compression more efficient. Especially algorithms relying on Block Motion Compensation already subdivide the image into 16×16 pixel regions that can then be either encoded using predictive coding

relying on the motion compensation or, if the motion search failed, without any prediction in its raw form. The decision is signaled to the decompression algorithm using a bit field containing one bit for each block.

Spectral redundancy reducing approaches rely on a wavelet transformation encoding the image in a single phase, whereas local approaches encode the image line- and columnwise and thus use different residual coding techniques. Because it is infeasable to combine different prediction schemes in that way, the use of selective predictions was discontinued in this thesis.

4 Results and Discussion



(a) *Shoe.* **(b)** *Head.*

(c) *Angel.* **(d)** *Constant.*

Figure 4.1: *Example pictures of the individual datasets I used to perform evaluations. The picture of the Angel dataset is the maximum image of all possible lighting directions.*

This chapter evaluates the individual compression approaches for multi-view and material capture on different datasets. The *Angel* dataset is a multi-view material capture dataset captured with the CultArc3D scanner of the CultLab3D project [San+14]. The

images are captured using the IC11000CU industry camera from NET GmbH with a resolution of 3840×2748 pixels and a bitdepth of 12 bit. The CultArc3D capturing device consist of two arcs that are each equipped with 9 cameras and 9 lights. The arcs are able to move in in one direction independently from each other with an amount of 9 steps, which results in 9×9 possible camera and 9×9 light source positions. The *Shoe* dataset containing 597 images was captured with the CultArm3D scanner [San+14] with an attached NIKON D610 that has a resolution of 6080×4028 pixels and a bitdepth of 14 bit. The *Head* dataset containing 252 images was also captured with the CultArm3D scanner, except for a Phase One A/S iXG 100MP with a resolution of $11\,608 \times 8708$ pixels and 16 bit attached. The *Constant* dataset containing 10 images was produced to evaluate inter-view redundancies and was captured with a Canon EOS 700D with a resolution of 5184×3456 and 14 bit. Example pictures from the datasets can be seen in fig. 4.1.

The original file format of NIKON, Canon and PhaseOne cameras are NEF, CR2 and IIQ. These formats are losslessly compressed with proprietary algorithms. The images of the NET camera must be read using an API that directly communicates with the camera, producing uncompressed 16 bit TIFF files that contain scaled 14 bit samples.

In this chapter, I evaluate different residual coding techniques in the first part and evaluate the individual compression approaches from chapter 3 on multi-view and material capture datasets in the second part. The multi-view and material capture datasets are evaluated separately on different datasets, as some of the approaches make assumptions about the input data. I also implemented a corresponding decompressing algorithm to verify that the compression is lossless and reversible.

4.1 Residual Coding

I compare several residual coding techniques on the output of different approaches. For this reason, I use the context-adaptive coder on the CDF 5/3 wavelet transformation (Section 3.5), an arithmetic coder with an dynamic frequency model (Section 2.4) and the Gzip algorithm (Section 2.11). The outcome is measured in *bits per pixel* (bpp) and compared against the raw format, which is in this case always a Bayer pattern of 16 bit.

To provide a wide variation of input data, I use the residual output of different prediction approaches. In one case I assume that the residual image is just the original plain image. The plain image and the LOPT-3D approach with prior BMC and normal BMC were tested on the Shoe dataset. All three approaches are fundamentally different and produce different residual images. The plain image still includes local and inter-view redundancies and the residual image has normal image statistics. The LOPT-3D approach with BMC

Table 4.1: *Comparison of the individual residual coding techniques against an uncompressed encoding. All values are measured in bits per pixel, the smallest value is marked in bold.*

Predictor	Dataset	Uncompressed	Spectral	Arith	Gzip
Uncompressed	Shoe	64	**30.417**	44.814	43.887
LOPT + BMC	Shoe	64	31.51	**30.973**	40.181
BMC	Shoe	64	**31.671**	33.288	42.155
LOPT (mat. capt.)	Angel	64	23.616	**22.879**	28.708
Delta	Constant	64	31.845	**30.842**	40.606

takes local and inter-view redundancies into account at the same time and thus creates very low frequency residual images that mainly contain noise. The BMC approach only takes inter-view redundancies into account. Residual images of BMC contain block-like structures caused by the underlying motion search blocks. The residual image has high frequencies on block boundaries. An evaluation of BMC in wavelet domain is not possible here, because the residual is already a wavelet tree and thus not suitable for most residual coders. The LOPT algorithm without BMC and the delta approach was performed on the Angel dataset. The delta approach depends on the previous image. If the images are similar, the residual image contains only noise as it removes almost all inter-view redundancies, which was tested on the constant dataset. If the images are different, the image statistics contain a wide frequency spectrum.

The comparison is shown in table 4.1 and fig. 4.2. It can be observed that the simple arithmetic coder does not perform well on natural images (i.e. images that are not residuals), resulting from the fact that they are not decorrelated. Gzip also does not perform much better on correlated images than the arithmetic coder, which means that the dictionary of Gzip adapts to the data very poorly. Gzip is anyway always outperformed by at east on other residual coder, i.e. it is never the best. Another observation is that the simple arithmetic coder works better on outputs of LOPT-3D and on the constant image difference. Values of these residual images mostly contain low frequencies meaning that the frequency spectrum is not big enough in contrast to natural images to be exploited by spectral approaches. The original image, the BMC prediction and the delta prediction have in common that their statistics behave more like natural images, which can be compressed well by wavelet compression techniques.

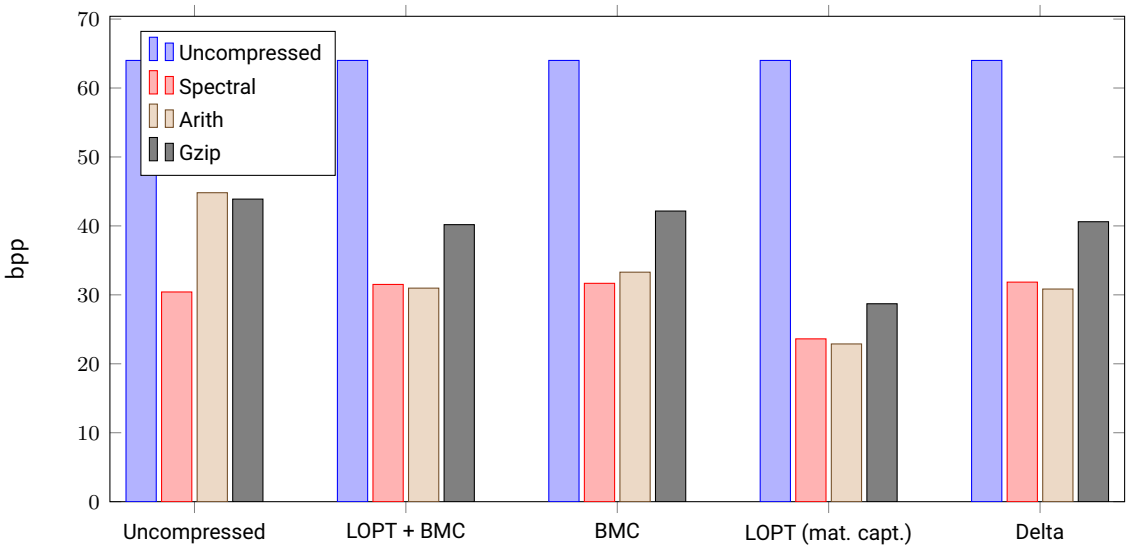


Figure 4.2: *Comparison of the individual residual coding techniques against an uncompressed encoding.*

4.2 Compression Rate

The main focus of this thesis is on efficient dataset compression for archiving purposes. In this section, I present and compare the compression rates that can be achieved by the individual approaches for both multi-view datasets, followed by the material capture dataset.

4.2.1 Geometry Reconstruction

In this section, I evaluate the individual approaches on the multi-view datasets Shoe and Head. As stated before, the Head dataset has a significantly higher image resolution than the Shoe dataset.

It can be seen in table 4.2 and fig. 4.3 that Gzip reduces the dataset size to 37.1 bpp or 42 % compared to an uncompressed encoding. The dataset size is further reduced by the Block Motion Compensation algorithm that removes only temporal redundancies to 33.1 bpp. The BMC algorithm in the wavelet domain performs better than standard BMC with a size of 31.5 bpp and is comparable to the results of the LOPT-3D with 31.3 bpp. In section 4.4 it can be observed that the computation time is significantly less for the wavelet BMC algorithm than for the LOPT-3D algorithm, making LOPT-3D a less attractive choice for compression of multi-view datasets. The spectral 2D approach performs best with a size of 30.6 bpp making it together with its computational simplicity the best suitable

Table 4.2: *Comparison of the individual multi-view compression approaches on the Shoe dataset.*

Approach	Bits per pixel	Total size in GB
Uncompressed	64	27.233
Gzip	37.087	15.781
PNG	41.737	17.76
2D	30.634	13.035
BMC	33.056	14.066
LOPT	31.347	13.339
Wavelet BMC	31.52	13.412

Table 4.3: *Comparison of the individual multi-view compression approaches on the Head dataset.*

Approach	Bits per pixel	Total size in GB
Uncompressed	64	47.447
Gzip	37.644	27.907
PNG	46.702	34.622
2D	36.13	26.786
BMC	37.552	27.839
LOPT	35.847	26.575
Wavelet BMC	39.194	29.057

approach for compressing this multi-view dataset.

Table 4.3 and fig. 4.4 show the results of the multi-view Head dataset, which are similar to the results of the Shoe dataset, except that the LOPT-3D approach has a slightly better (less than 1 %) compression rate than the 2D spectral approach there. The results of the Head dataset generally require 88 % more bits per pixel as the Shoe dataset. This is expected as the input data has with 16 bit a higher bit depth compared to 14 bit for the Shoe dataset. This dataset can be compressed using the LOPT algorithm with a size of 35.847 bpp, which is comparable to the result of the 2D spectral approach with 36.13 bpp.

Summarizing, the two-dimensional spectral approach reduces the size with rates from 1.77:1 to 2.09:1 compared to an uncompressed encoding and with rates between 1.29:1 to 1.36:1 compared to PNG.

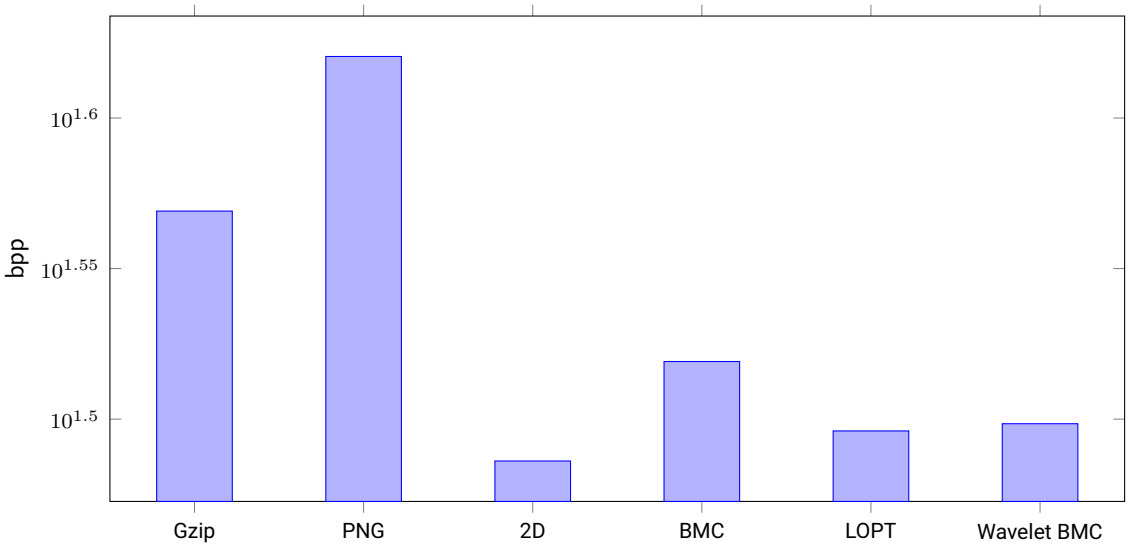


Figure 4.3: *Comparison of the individual multi-view compression approaches on the Shoe dataset. The axis is in logarithmic scale.*

4.2.2 Material Capture

Material capture approaches were evaluated on the Angel dataset. I used the images from that dataset captured from one constant camera postion. Compression across the different camera positions is equal to multi-view compression (e.g. of the maximum image) and not further evaluated in this thesis.

It can be clearly observed from fig. 4.5 that the 4D wavelet approach produces poor results. This indicates that there are no spectral redundancies between views. It can also be observed that the CDF 9/7 approach produces even worse results. Reasons for this might be that the composition of individual steps of this approach introduce further entropy to the residual image caused by artifacts in high frequency areas. High frequency changes can only be compensated poorly with the multiplication of the factor image. This indicates that complex algorithms, which produce visually pleasing predictions, are often not useful for lossless compression. The prediction produced by the CDF 9/7 approach is also an insufficient input for the local LOPT approach. However, the subsequent LOPT step reduces the size by almost 25 %. Table 4.4 indicates that the central difference is the best input for LOPT-3D and that it performs better than the average and the maximum. The LOPT approach is still slightly better than the two-dimensional spectral approach but the computational effort of LOPT is disproportionately greater to the relative difference of 0.1 %. Compared to an uncompressed format, the file size can be reduced with compression rates of 2.75:1 and compared to PNG with 1.33:1. In contrast to both multi-view datasets,

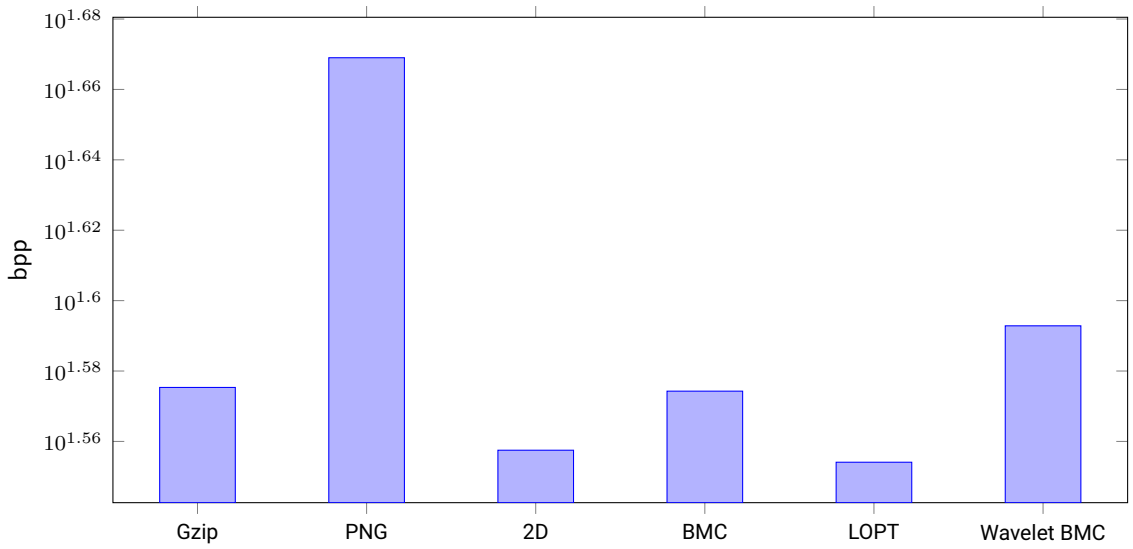


Figure 4.4: *Comparison of the individual multi-view compression approaches on the Head dataset. The axis is in logarithmic scale.*

PNG performs better than Gzip for this dataset. The rates between PNG and the 2D spectral approach have similar values between all datasets.

4.3 Theoretical Considerations

In this section, I evaluate the datasets under theoretical aspects, to get a depper understanding, why some compression approaches presented in chapter 3 or general purpose compression algorithms do not perform as expected.

4.3.1 Optimal Dataset

Because the rates of inter-view redundancy reduction were smaller than expected, I evaluated well performing approaches on a near-optimal dataset (the Constant dataset). This dataset was captured using a tripod capturing a nearly motion-free and constant illuminated scene with fixed camera position and rotation. This dataset should have maximal temporal redundancy, except for different types of noise caused by the digital camera that cannot be prevented.

For this evaluation, I introduce a further simple compression approach that predicts the current frame using its parent and only its parent and encodes it using predictive coding and a simple arithmetic coder as in section 4.1. The second approach I use in this section is the local LOPT-3D approach with a prior BMC step and a search space of 10 pixels

Table 4.4: *Comparison of the individual material capture compression approaches.*

Approach	Bits per pixel	Total size in MB
Uncompressed	64	1630.283
Gzip	31.947	813.791
PNG	29.068	740.461
2D	21.847	556.506
LOPT cen	21.794	555.986
LOPT avg	22.002	560.47
LOPT max	22.372	569.896
CDF97	28.814	733.99
LOPT + CDF97	22.668	577.417
4D	25.317	644.894

Table 4.5: *Comparison of different approaches on the Constant dataset.*

Approach	Bits per pixel	Total size in MB
Uncompressed	64	341.719
Delta	31.52	174.986
Local	30.405	168.792
Spectral	30.502	169.331
CR2	38.705	214.869

to compensate remaining motion caused by small vibrations. The last approach is the two-dimensional spectral compression algorithm described in section 3.4, which ignores all inter-view redundancies and encodes each view independently. For completeness, I also compared it to the Canon CR2 raw format.

As it can be seen in table 4.5 and fig. 4.6, the compression rates of the delta approach are inferior to rates of the local and spectral ones. This indicates fewer temporal redundancies as expected and that the dataset contains too much noise that does not correlate across views. Each image consist of the ground truth data with uncorrelated and correlated mean-free noise. Thus, the difference between two images has twice the amount of noise. As long as a compressor is better at encoding the ground truth signal than the noise, a prediction based on a single noisy input is going to be worse. Similar to the material capture compression results, the local approach performs slightly better than the spectral approach but does still not justify the computational effort. The reason for this is the following. The

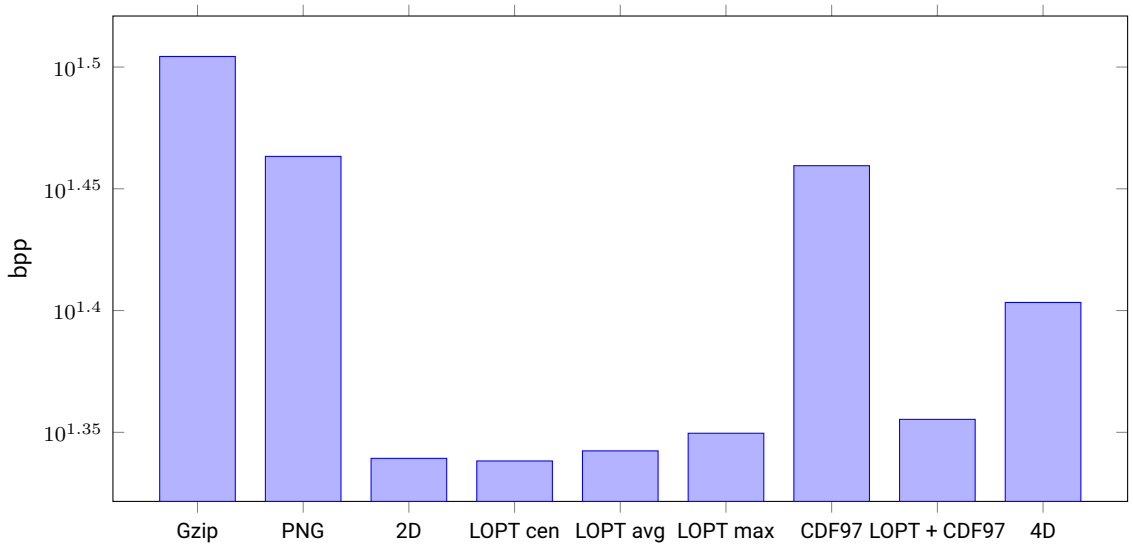


Figure 4.5: *Comparison of the individual material capture compression approaches. The axis is in logarithmic scale.*

LOPT approach with center prediction smoothes out noise and thus performs better.

4.3.2 Gzip on Natural Images

As Gzip is often used to losslessly compress natural image data, I evaluate the compression rate of Gzip and a simple arithmetic coder on a set of images from the datasets from above. I additionally compress an image *Phone* captured using a Smartphone camera and an additional monochrome image *JPEG* that was taken using a DLSR camera and originally saved in the JPEG file format.

As it can be clearly observed from table 4.6 and fig. 4.7, Gzip performs approximately

Table 4.6: *Comparison of Gzip and a simple arithmetic coder. All values are measured in bits per pixel, smaller values are marked in bold.*

Dataset	Gzip	Arith
Constant	44.429	**42.641**
Angel	7.55	**6.065**
Head	47.187	**46.26**
Shoe	**44.022**	45.068
Phone	**10.113**	21.947
JPEG	**4.214**	9.005

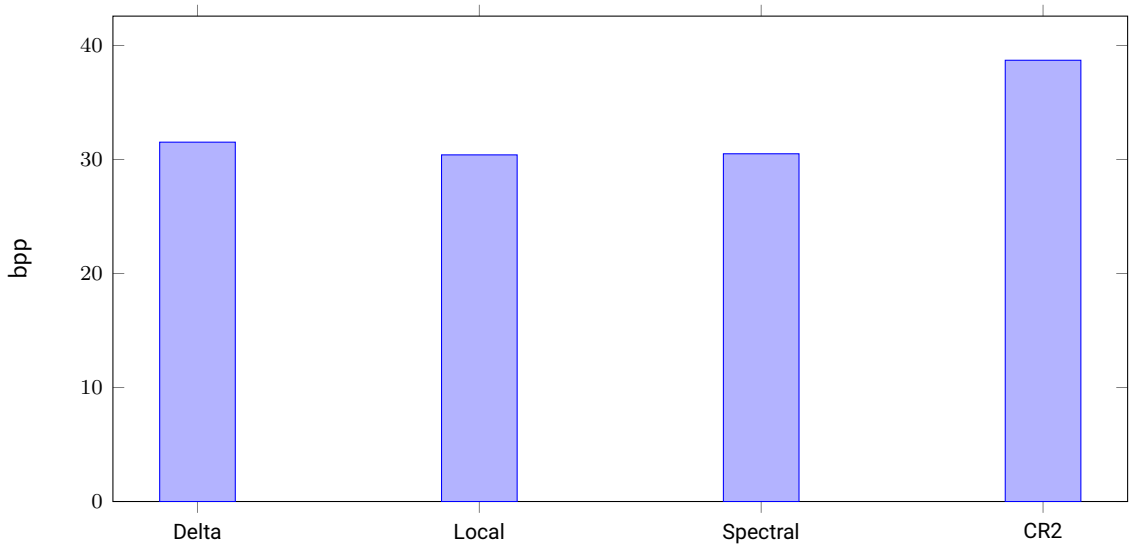


Figure 4.6: *Comparison of different approaches on the Constant dataset.*

equally as good as the arithmetic coder for the datasets originating from raw file formats. For the most datasets, the arithmetic coder also performs better than Gzip. Images stored as JPEG files or that go through various image optimization steps, tend be compressed twice as good as using Gzip with the arithmetic coder, because the JPEG algorithm produces only a limited number of pixel values within an image row (i.e. *artificial redundancy*). Because Gzip uses a dictionary lookup with a followup coding of characters and offsets using an entropy coder, this comparison with a standalone entropy coder indicates that the dictionary is almost not used during compression. Thus, compression with Gzip is not suitable for multi-view datasets, material capture datasets and generally natural images.

4.3.3 Analysis of the Arithmetic Coder Models

In this section the actual utilization of the models of the arithmetic coder is analyzed on a natural image. Figure 4.8 shows frequency plots of the models for the first and second context selection. As the CDF 9/7 wavelet transform effectively computes image derivatives, subbands are expected to be heavy tailed [WS00]. This expectation is confirmed by the plot. Figure 4.8(a) and fig. 4.8(b) show similar results, because the LSB of the second context selection is blurred out. Additionally it is observable that fig. 4.8(b) shows higher frequencies than fig. 4.8(a), which is also expected, because the finest wavelet subband is missing in the first context selection as it is no parent of another subband.

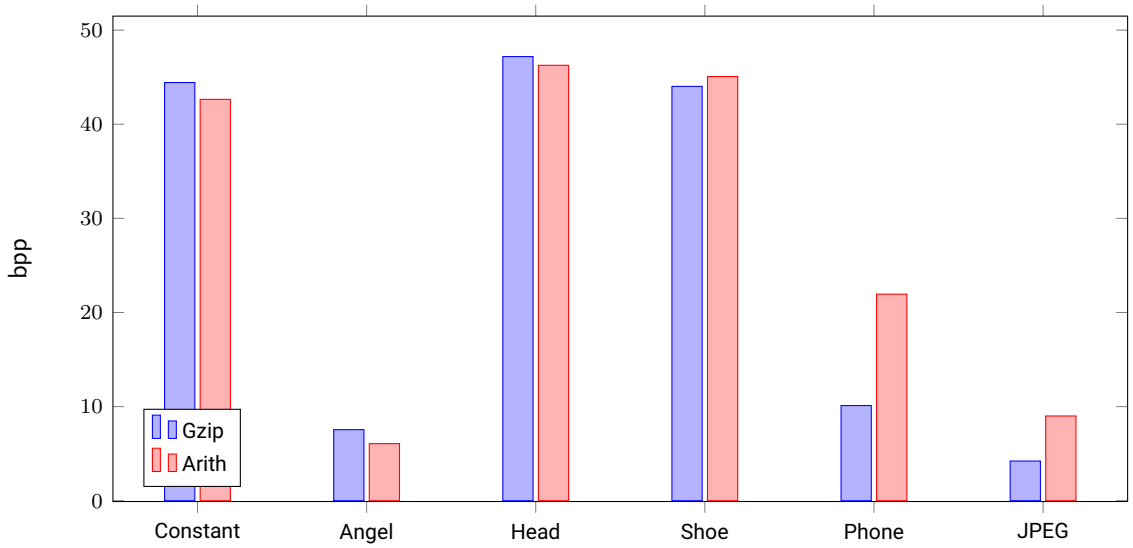


Figure 4.7: *Comparison of Gzip and a simple arithmetic coder.*

4.4 Run-Time Performance

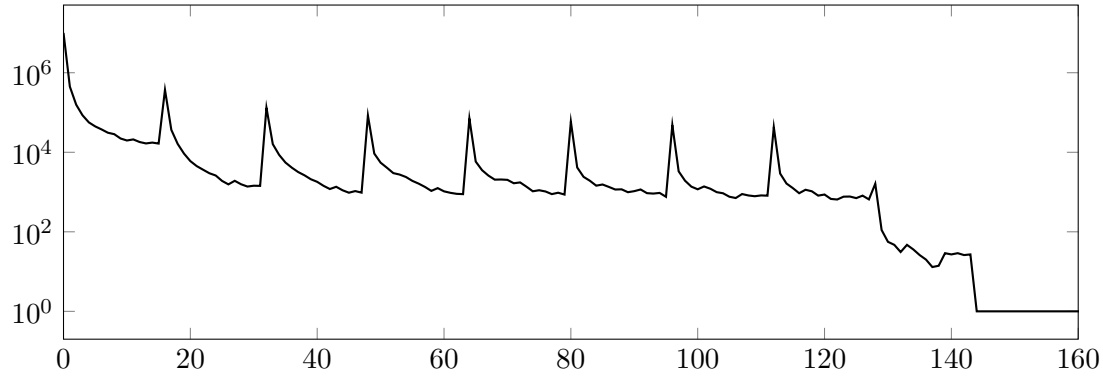
Even though run-time performance is only a secondary concern for archiving purposes, it was measured it in order to determine the computational effort of each approach. Execution times are measured on an Intel Xeon E5-2650 v2 CPU.

Figure 4.9 shows that the computation time of PNG depends on the dataset. For the multi-view approaches, Gzip is besides of PNG the best performing algorithm. It is also clearly observable that all approaches that require the LOPT-3D or the BMC algorithm have the highest run-time.

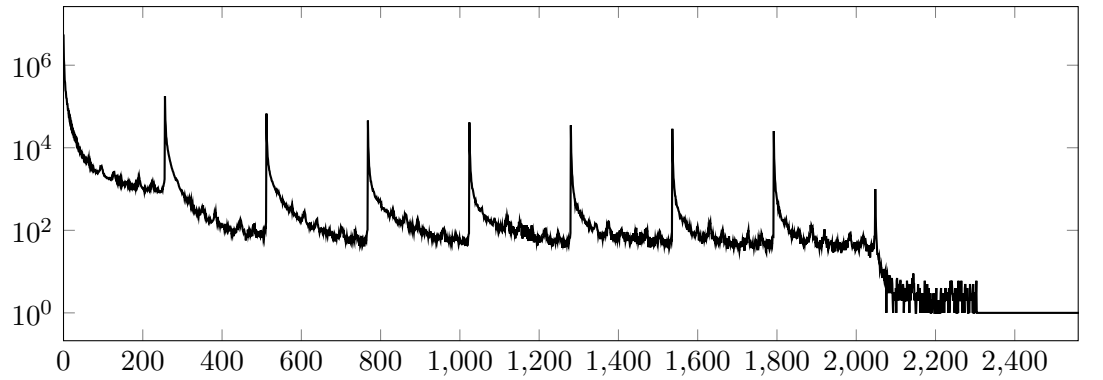
PNG requires approximately 4 min to compress the multi-view Shoe dataset, followed by Gzip which require 11 min. The 2D spectral approach requires 1 h. The wavelet BMC approach requires a computational expensive motion search and thus takes 17 h to complete, even though it is accelerated by dividing it into two steps. The standard BMC approach takes with more than 3 d significantly more time. The succeeding LOPT-3D step takes almost 22 h resulting in more than 4 d to complete.

The multi-view Head dataset has higher resolution input images and thus require 18 min to complete the Gzip compression, followed by the 2D spectral approach with 111 min. For this dataset, PNG requires with 135 min more time than Gzip and the 2D approach. The BMC approach in wavelet domain requires 30 h to complete, followed by the BMC approach requiring 2 d and the LOPT-3D approach requiring 3 d.

Gzip, PNG and the spectral 2D approach perform similar for multi-view approaches



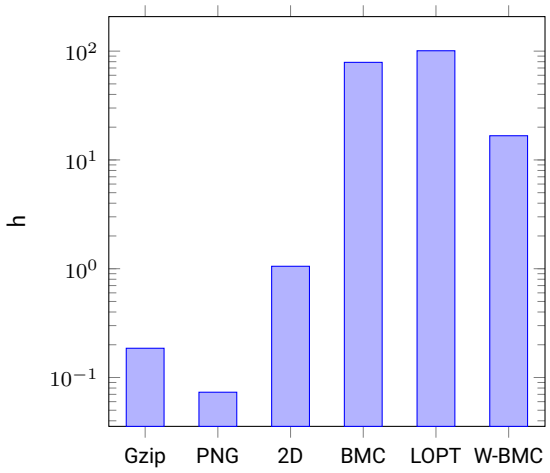
(a) *Utilization of the* 16 *models of the first context selection that are chosen based on the MSB of the parent coefficient and captures the statistics for the MSB of the current coefficient. The frequency tables of the* 16 *models are concatenated in this plot. The values from* 161 *to* 255 *remained* 1 *(the intial frequency) and are left out in this plot.*



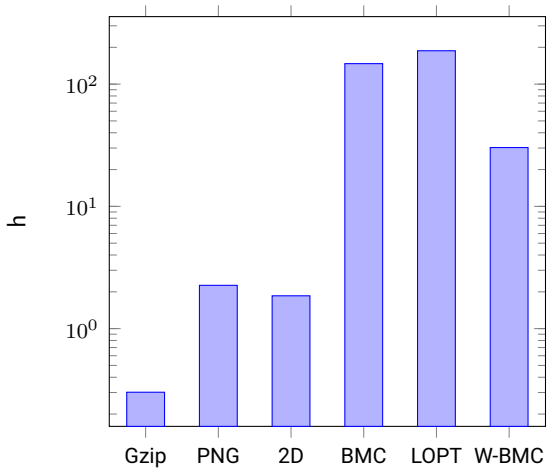
(b) *Utilization of the* 256 *models of the second context selection that are chosen based on the MSB of the current coefficient and captures the stated for the LSB of it. Again, values with the frequency tables are concatenated and values with initial frequencies are left out.*

Figure 4.8: *Model utilization of the arithmetic coder. All plots are in logarithmic scale.*

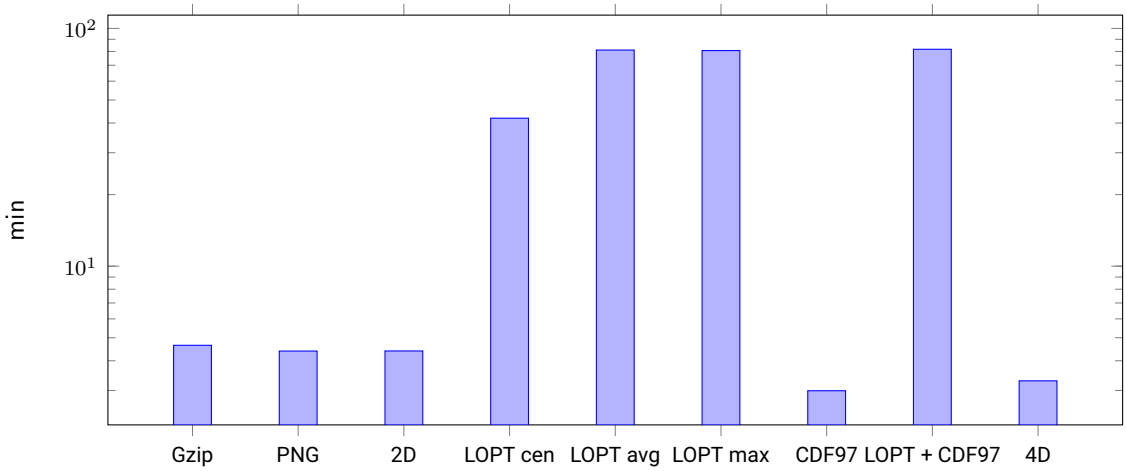
on the Shoe dataset compared to material capture approaches all requiring approximately 4 min to complete, neglecting noise due to the smaller dataset size. The CDF 9/7 approach has slightly less time consumption but it is less attractive due to its inferior compression rate. The 4D spectral approach has similar time consumptions as the 2D approach. The LOPT-3D approach with mean prediction requires with 42 min approximately halve the time than all other LOPT approaches which require approximately 81 min to complete.



(a) *Run-time of the individual multi-view compression approaches on the Shoe dataset.*



(b) *Run-time of the individual multi-view compression approaches on the Head dataset.*



(c) *Run-time of the individual material capture compression approaches on the Angel dataset.*

Figure 4.9: *Overall run-time performance. Axis of all plots are in logarithmic scale.*

5 Conclusion

In this thesis I presented lossless approaches for compression of structured and unstructured multi-view and material capture datasets with focus on data from cultural heritage digitalization. These approaches target spectral, local and temporal redundancies and work on arbitrary image resolutions and bitdepths.

The evaluation showed that residual images (e.g. those of the LOPT-3D algorithms) do contain few to no spectral redundancies anymore. This can be explained by the context-adaptive coding using the wavelet tree already decorrelating the data well enough and the violation of the natural image property of the wavelet tree. Thus, mixing up local with spectral approaches leads to worse results than coding the residual images directly using an entropy coder. Unfortunately, the compression rates multi-dimensional spectral approaches were inferior as it is not possible to perform a prior brightness or motion compensation step to find correlating regions between pairs of images enlarging spectral inter-view redundancies.

However, the LOPT-3D approach working on means of adjacent image pairs, reducing local and temporal redundancies, has slightly better compression rates than the two-dimensional spectral approach for material capture datasets. Nevertheless, the computational effort required by the LOPT-3D algorithm to compute the optimal weighting and the prior BMC step for multi-view datasets is disproportional to the storage savings. For multi-view datasets, the LOPT-3D approach performs significantly worse, which indicates that camera baselines are too wide to find correlations between images. Other approaches that address only temporal redundancies, like the BMC view prediction in the spatial and wavelet domain and the CDF 9/7 material capture prediction, perform generally worse than local and spectral approaches.

Theoretical experiments on an “optimal” dataset unveiled that even if it contains no visible difference between the views, the compression rate results are similar to real world datasets, indicating that noise is too high compared to temporal redundancies. This makes it hard to achieve better compression rates with local redundancy reduction compared to spectral approaches. As the LOPT-3D and the 2D spectral approach lead to similar results

and LOPT further incorporates temporal redundancies, it indicates that the images contain fewer local than spectral redundancies. Gzip tends to achieve similar results on the datasets as a simple arithmetic coder, meaning that the dictionary utilization is inefficient.

Moreover, temporal approaches prevent random access to individual images of the datasets as all frames depend on each other and thus need to be decoded sequentially. Random access can be easily enabled for the two-dimensional spectral approach by coding a index structure prior to the actual data. Results show, that algorithms that achieve better results tend to consume significantly more time, which can be explained by the complexity of compression algorithms [Kol65; Say02].

Summarizing, the two-dimensional spectral approach has the best balance between compression rate and running time for compression of structured and unstructured multi-view and material capture datasets. Compared to an uncompressed encoding, it achieves compression rates between 1.77:1 to 2.75:1 for both multi-view datasets and 2.75:1 for the material capture dataset. Compared to the PNG algorithm, it achieves rates of 1.33:1 on average on all datasets. Compression with Gzip results in very inconsistent rates.

5.1 Future Work

As described above, the local approach using the LOPT-3D algorithm tends to achieve slightly better results than the two-dimensional spectral approach. To further improve the results, it would be interesting to know how multi-BMC [CD05] influences the result and how many views are sufficient.

Targeting spectral approaches, further research can be done to find correlating regions to perform high dimensional wavelet transformations and to encode them context-adaptively. This would be a lossless refinement of the layer-based approach of Gelman et al. [GDV10] taking advantage of the good performing and low-complexity spectral algorithms. However, the compression are not expected to be optimal, because our tests on a constant dataset showed that lossless temporal compression is generally inefficient.

In regard to lossy compression near-lossless algorithms would be an interesting alternative to lossless compression. These near-lossless approaches could be driven by certain constraints, e.g. that image feature descriptors must be similar up to a certain threshold compared to the original version. Lossy approaches would help to propagate the datasets to further storage locations, which contributes to further preservation of cultural heritage. Other ideas would be to perform a prior masking step removing all parts of the image that are not required for further processing (e.g. the background and parts of the capturing environment that also appear in the background).

Multi-view reconstructions are also captured beyond cultural heritage digitalization. Datasets captured to create digital copies for use in computer games, animations, 3D printing or similar purposes, potentially created from community photo collections [Goe+07] are often not stored in the original raw image format in favor to the JPEG image format. However, re-encoding the lossy JPEG compressed datasets with a lossless approach would increase storage size. Developing a lossy algorithm with additional inter-view redundancy reduction that is able to decompress with similar visual results as JPEG would be promising alternative to a sequential JPEG encoding. Temporal redundancies are generally higher in lossy environments.

Community photo collections usually contain images with different resolutions. In terms of compatibility, it would be useful to be able to compress datasets with inconsistent resolutions and bitdepths. Additionally, a prior encoding of camera paramters (e.g. color transform matrices or white balance histograms) would be useful, to generate final visualizations of the decompressed images.

Bibliography

[Jar30] Vojtěch Jarník. “O jistém problému minimálním”. In: *Práca Moravské Prírodovedecké Spolecnosti* 6 (1930), pp. 57–63.

[Huf52] David A Huffman. “A method for the construction of minimum-redundancy codes”. In: *Proceedings of the IRE* 40.9 (1952), pp. 1098–1101.

[Eli55] Peter Elias. “Predictive coding–I”. In: *IRE Transactions on Information Theory* 1.1 (1955), pp. 16–24.

[Kol65] Andrei N Kolmogorov. “Three approaches to the quantitative definition of information”. In: *Problems of information transmission* 1.1 (1965), pp. 1–7.

[ZL77] Jacob Ziv and Abraham Lempel. “A universal algorithm for sequential data compression”. In: *IEEE Transactions on information theory* 23.3 (1977), pp. 337–343.

[RL79] Jorma Rissanen and Glen G Langdon. “Arithmetic coding”. In: *IBM Journal of research and development* 23.2 (1979), pp. 149–162.

[AB91] Edward H. Adelson and James R. Bergen. “The plenoptic function and the elements of early vision”. In: *Computational Models of Visual Processing*. MIT Press, 1991, pp. 3–20.

[Pae91] Alan W Paeth. “Image file compression made easy”. In: *Graphics Gems II*. Elsevier, 1991, pp. 93–100.

[CDF92] Albert Cohen, Ingrid Daubechies, and J-C Feauveau. “Biorthogonal bases of compactly supported wavelets”. In: *Communications on pure and applied mathematics* 45.5 (1992), pp. 485–560.

[Fen93] Peter Fenwick. “A new data structure for cumulative probability tables”. In: *Software-Practice and Experience* (1993).

[Sha93] Jerome M Shapiro. “Embedded image coding using zerotrees of wavelet coefficients”. In: *IEEE Transactions on signal processing* 41.12 (1993), pp. 3445–3462.

[AKH95] Halûk Aydinoglu, Faouzi Kossentini, and Monson HIII Hayes. “A new framework for multi-view image coding”. In: *1995 International Conference on Acoustics, Speech, and Signal Processing*. Vol. 4. IEEE. 1995, pp. 2173–2176.

[Fen95] Peter Fenwick. *A New Data sturcture for cumulative Probability Tables: an Improved Frequency to Symbol Algorithm*. Tech. rep. Department of Computer Science, The University of Auckland, New Zealand, 1995.

[Col96] Robert T Collins. “A space-sweep approach to true multi-image matching”. In: *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR’96, 1996 IEEE Computer Society Conference on*. IEEE. 1996, pp. 358–363.

[WSS96] Marcelo J Weinberger, Gadiel Seroussi, and Guillermo Sapiro. “LOCO-I: A low complexity, context-based, lossless image compression algorithm”. In: *Proceedings of Data Compression Conference-DCC’96*. IEEE. 1996, pp. 140–149.

[Bou97] Thomas Boutell. *PNG (portable network graphics) specification version 1.0*. Tech. rep. 1997.

[Laf+97] Eric PF Lafortune et al. “Non-linear approximation of reflectance functions”. In: *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co. 1997, pp. 117–126.

[Sie+97] Mel Siegel et al. “Compression and interpolation of 3d stereoscopic and multiview video”. In: *Stereoscopic Displays and Virtual Reality Systems IV*. Vol. 3012. International Society for Optics and Photonics. 1997, pp. 227–239.

[MF98] Bo Martins and Søren Forchhammer. “Lossless compression of video using motion compensation”. In: *Proceedings DCC’98 Data Compression Conference (Cat. No. 98TB100225)*. IEEE. 1998, p. 560.

[MNW98] Alistair Moffat, Radford M Neal, and Ian H Witten. “Arithmetic coding revisited”. In: *ACM Transactions on Information Systems (TOIS)* 16.3 (1998), pp. 256–294.

[WS00] Martin J Wainwright and Eero P Simoncelli. “Scale mixtures of Gaussians and the statistics of natural images”. In: *Advances in neural information processing systems*. 2000, pp. 855–861.

[SD01] Druti Shah and Neil A Dodgson. “Issues in multiview autostereoscopic image compression”. In: *Stereoscopic Displays and Virtual Reality Systems VIII*. Vol. 4297. International Society for Optics and Photonics. 2001, pp. 307–317.

[Sha01] C. E. Shannon. “A Mathematical Theory of Communication”. In: *SIGMOBILE Mob. Comput. Commun. Rev.* 5.1 (Jan. 2001), pp. 3–55.

[Bru+02] Dania Brunello et al. “Lossless video coding using optimal 3D prediction.” In: *ICIP (1)*. 2002, pp. 89–92.

[GP02] Larisa Goffman-Vinopal and Moshe Porat. “Color image compression using inter-color correlation”. In: *Proceedings. International Conference on Image Processing*. Vol. 2. IEEE. 2002, pp. II–II.

[Say02] Khalid Sayood. *Lossless compression handbook*. Elsevier, 2002.

[KJ03] Chaitanya Kamisetty and CV Jawahar. “Multiview image compression using algebraic constraints”. In: *TENCON 2003. Conference on Convergent Technologies for Asia-Pacific Region*. Vol. 3. IEEE. 2003, pp. 927–931.

[MS03] Henrique Malvar and Gary Sullivan. “YCoCg-R: A color space with RGB reversibility and low dynamic range”. In: *ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q* 6 (2003).

[Gut04] Stefan Guthe. “Compression and visualization of large and animated volume data”. PhD thesis. University of Tübingen, Germany, 2004.

[CD05] Elias SG Carotti and Juan Carlos De Martin. “Motion-compensated lossless video coding in the CALIC framework”. In: *Proceedings of the Fifth IEEE International Symposium on Signal Processing and Information Technology, 2005.* IEEE. 2005, pp. 600–605.

[GD07] Nicolas Gehrig and Pier Luigi Dragotti. “Distributed compression of multi-view images using a geometrical coding approach”. In: *2007 IEEE International Conference on Image Processing*. Vol. 6. IEEE. 2007, pp. VI–421.

[Goe+07] Michael Goesele et al. “Multi-view stereo for community photo collections”. In: *2007 IEEE 11th International Conference on Computer Vision*. IEEE. 2007, pp. 1–8.

[BVL10] Benjamin Battin, Philippe Vautrot, and Laurent Lucas. “A new near-lossless scheme for multiview image compression”. In: *Stereoscopic Displays and Applications XXI*. Vol. 7524. International Society for Optics and Photonics. 2010, 75241P.

[GDV10] Andriy Gelman, Pier Luigi Dragotti, and Vladan Velisavljević. “Multiview image compression using a layer-based representation”. In: *2010 IEEE International Conference on Image Processing*. IEEE. 2010, pp. 493–496.

[Vel+11] Vladan Velisavljević et al. “View and rate scalable multiview image coding with depth-image-based rendering”. In: *2011 17th International Conference on Digital Signal Processing (DSP)*. IEEE. 2011, pp. 1–8.

[San+14] Pedro Santos et al. “CultLab3D - On the Verge of 3D Mass Digitization”. In: *Eurographics Workshop on Graphics and Cultural Heritage*. Ed. by Reinhard Klein and Pedro Santos. The Eurographics Association, 2014.

[Per15] Cristian Perra. “Lossless plenoptic image compression using adaptive block differential prediction”. In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2015, pp. 1231–1234.

[PA16] Cristian Perra and Pedro Assuncao. “High efficiency coding of light field images based on tiling and pseudo-temporal data arrangement”. In: *2016 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*. IEEE. 2016, pp. 1–4.